UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

PROYECTO FIN DE CARRERA

Ingeniería Técnica Industrial, Electricidad

Interfaz Gráfica Para La Simulación De Modelos Dinámicos

AUTOR: Santiago Galán Morales
TUTOR: Joaquín-Eloy García Carrasco

Leganes, 07 de Diciembre de 2011

ÍNDICE

RESUMEN DEL PROYECTO

En este proyecto se ha realizado el control de cualquier sistema basado en la herramienta Simulink a través de la herramienta Guide. Con la realización de un interfaz gráfico de usuario, podemos modificar cualquier parámetro del sistema creado en Simulink. Con este GUI, no solo modificamos los parámetros si no que además nos permite guardar los datos y los cambios en el sistema Simulink.

Con una programación genérica y adaptable a cualquier modelo de Simulink de una forma común. En este proyecto se ha particularizado para un modelo concreto, en el que se simulan distintos métodos de arranque de máquinas de inducción.

La interfaz se ha programado orientada a dos posibles usos. Por un lado un usuario básico que sólo pretende usar las funciones disponibles sin modificar parámetros no visibles, y por otro, un usuario experto que podrá acceder prácticamente a todos los parámetros disponibles en un modelo de Simulink para modificar el comportamiento de la simulación tanto como quiera desde la GUI.

Este GUI contiene una programación de unas 9000 líneas, claras y explicadas para su posterior estudio.

Este proyecto también ha consistido en el estudio de la simulación dinámica del arranque de un motor de inducción, tanto en su parte teórica como práctica. Su parte teórica consiste en el estudio de las ecuaciones de un motor de inducción así como los teoremas de Park y Clarke. Su parte práctica ha consistido en el estudio del sistema Simulink de simulación dinámica del arranque de un motor de inducción.

En esta redacción hemos querido exponer el proyecto así como una introducción al Matlab y al Simulink. También hemos querido explicar lo que es una interfaz gráfica para el usuario.

CAPÍTULO 1. INTRODUCCIÓN AL MATLAB. [1], [5], [6]

Matlab (Matrix Laboratory, “laboratorio de matrices”).

Como información general podemos mencionar que Matlab tiene un modelo de desarrollo de Software propietario, de genero Software matemático lazado en 1984.

Este software de ayuda al cálculo está formado por un lenguaje de programación propio (lenguaje M). Entre sus aplicaciones se hallan: el cálculo matemático de matrices, manipulación y figuración de datos y funciones, el desarrollo de algoritmos, la realización de interfaces de usuario (GUI‘s), un entorno de programación visual (Simulink) y la comunicación con programas en otros lenguajes y con otros dispositivos hardware. Está disponible para los sistemas operativos Windows, Unix y Apple Mac.

Matlab dispone de dos aplicaciones adicionales que amplían sus propiedades, que son: Simulink (plataforma de simulación) y GUIDE (editor de interfaces de usuario - GUI). Además, se pueden aumentar las características de Matlab con las herramientas (toolboxes), y las de Simulink con las aplicaciones de bloques (blocksets).

Este software es utilizado en centros de desarrollo, centros de investigación, universidades y en empresas de i+d. En estos años se ha ampliado el número de prestaciones, como la de programar directamente procesadores digitales de señal o crear código VHDL.

1.1. Acotaciones

Al ser Matlab un software bajo patente de The Mathworks, hubo muchas críticas desde los usuarios, ya que estos estaban sujetos ha este. Actualmente se ha elaborado una aplicación "Application Deployment" para utilizar funciones Matlab como archivos de biblioteca que pueden ser usados de aplicación .net o Java.

1.2. Software‘s similares

Labview:

Labview es un entorno de programación gráfica usado para desarrollar sistemas de medida, pruebas y control. Utilizando íconos gráficos y diagramas de flujos. Cuya finalidad es ofrecer una integración de los dispositivos de hardware utilizando bibliotecas integradas para análisis avanzado y visualización de datos.

Como características principales podemos nombrar su programación rápida, liberada de código gracias a su forma de programar a través de bloques, su visuallización de datos e interfaces de usuario, su integración de harware, su almacenamiento de datos y reportes y sus comunicaciones.

CAPÍTULO 2. INTRODUCCIÓN AL SIMULINK. [1] [2]

Simulink es un entorno de programación visual, que funciona sobre el entorno de programación Matlab. De un alto nivel de programación con lenguaje interpretado por Matlab (archivos con extensión .m). Simulink genera archivos con extensión .mdl (de "model").

Simulink viene a ser una herramienta de simulación de modelos o sistemas, con cierto grado de abstracción de los fenómenos físicos involucrados en los mismos. Se hace hincapié en el análisis de sucesos, a través de la realización de sistemas (cajas negras que realizan alguna operación).

Se emplea en ingeniería electrónica en temas relacionados con el procesamiento digital de señales (DSP), involucrando temas específicos de ingeniería biomédica, telecomunicaciones, entre otros. También es muy utilizado en ingeniería de control y robótica. [1].

Simulink es una plataforma para simulación multidominio y diseño basado en modelos de sistemas dinámicos. Proporciona un entorno gráfico interactivo y un conjunto de librerías de bloques personalizables que permiten diseñar, simular, implementar y probar una gran variedad de sistemas con variación temporal, entre los que se incluyen sistemas de comunicaciones, control, procesado de señales, vídeo e imagen.

2.1. Funciones principales

- Bibliotecas extensas y ampliables de bloques predefinidos
- Editor de gráficos interactivos para ensamblar y administrar diagramas de bloque intuitivos
- Capacidad de gestionar diseños completos segmentando los modelos en jerarquías de componentes de diseño
- Explorer, para navegar, crear, configurar y buscar todas las señales, parámetros, propiedades y código generado asociados con el modelo
- Interfaces de programación de aplicaciones (API) que permiten conectar con otros programas de simulación e incorporar código escrito manualmente
- Embedded Matlab, Bloques de funciones para implementar los algoritmos de Matlab en Simulink e implementaciones de sistemas empotrados
- Modos de simulación (normal, acelerador y acelerador rápido) para ejecutar simulaciones de forma interpretativa o a velocidades de código C compilado
- Depurador y perfilador gráfico para examinar los resultados de simulación y diagnosticar el rendimiento y el comportamiento inesperado del diseño
- Acceso completo a Matlab para analizar y visualizar resultados, personalizar el entorno de modelaje y definir señales, parámetros y datos de prueba
- Herramientas de análisis de modelos y diagnosis para garantizar la coherencia de los modelos e identificar errores de modelaje

2.2. Crear y trabajar con los modelos

Con Simulink se puede crear, modelar y hacer el mantenimiento de un diagrama de bloques del sistema detallado utilizando un amplio conjunto de bloques predefinidos. Simulink cuenta con herramientas para el modelaje jerárquico, la gestión de datos y la personalización de subsistemas, por lo que resulta fácil crear representaciones concisas y precisas independientemente de la complejidad del sistema con el que se trabaja.

2.3. Selección y personalización de bloques

El software de Simulink incluye una amplia biblioteca de funciones que se usan habitualmente en el modelaje de un sistema. Estas incluyen:

- Bloques dinámicos continuos y discretos, como el de integración y el de retardo de la unidad
- Bloques de algoritmos, como el de suma, el de producto y el de tabla de búsqueda
- Bloques estructurales, como MUX, interruptor y selector de bus
- Pueden personalizarse los bloques integrados o crear nuevos bloques directamente en Simulink e introducirlos en las bibliotecas propias
- Definición y control de señales y parámetros

Simulink permite definir y controlar los atributos de las señales y parámetros asociados con el modelo. Las señales son cantidades que varían según el tiempo representadas por los bloques de conexión de líneas. Los parámetros son coeficientes que ayudan a definir la dinámica y el comportamiento del sistema.

Los atributos de las señales y parámetros pueden especificarse directamente en el diagrama o en un diccionario de datos independiente. Con el Model Explorer se puede gestionar el diccionario de datos y reorientar un modelo de forma rápida incorporando distintos conjuntos de datos.

2.4. Ejecución de una simulación

Después de crear el modelo con Simulink, puede simularse su comportamiento dinámico y comprobar los resultados en tiempo real. El software de Simulink ofrece varias funciones y herramientas para garantizar la velocidad y la precisión de la simulación, como solucionadores de paso fijo y de paso variable, un depurador gráfico.

2.5. Análisis de resultados

Simulink incluye varias herramientas para analizar el sistema, visualizando los resultados y probando, validando y documentando los modelos.

2.6. Visualización de resultados

Puede visualizarse el sistema observando las señales con las pantallas y vistas que ofrece el software de Simulink. Si no, también pueden diseñarse pantallas personalizadas con las herramientas de visualización y desarrollo de GUI de Matlab. También pueden registrarse las señales para el postproceso.

CAPÍTULO 3 ¿QUÉ ES UN GUI?

3.1. Introducción

GUI (Graphical User Interface), (Interfaz Gráfica de Usuario), es un entorno trabajo para la programación visual que nos proporciona Matlab para elaborar y ejecutar programas que necesiten ingreso continuo de datos. Tiene la propiedad básica de todos los programas visuales como Visual Basic o Visual C++.

Es una herramienta de trabajo que se extiende en el soporte de Matlab, planificada para crear interfaces gráficas para el usuario fácil y rápidamente, dando respaldo al diseño y presentación de los elementos de control de la interfaz, disminuyendo el esfuerzo al nivel de seleccionar, tirar, arrastrar y personalizar propiedades.

Una vez que los elementos están colocados de una manera visual aceptada en el GUI del archivo (*.fig), se editan las funciones de llamada (Callback) de cada uno de los elementos en el archivo GUI (*.m). Escribiendo el código de Matlab en él, se ejecuta cuando el elemento sea utilizado. Siempre será difícil diseñar GUI ‘s, pero no debería ser difícil implementarlas.

GUI esta diseñado para tener que ofrecer menos esfuerzo en el proceso de aplicación de la interfaz gráfica y obviamente para trabajar como herramienta de trazado de GUI‘s.

En el diseño de una GUI es muy importante el editor de propiedades (property editor). Entre sus componentes se encuentra disponible en cualquier momento que se esté trabajando con los controles de Matlab. El editor de propiedades se puede concebir como una herramienta de trazado y asistente de codificación (revisión de nombres y valores de propiedades). Cuando se fusiona con el panel de control, el editor de menú, y herramienta de alineación, resulta el control de los gráficos en Matlab.

El concepto básico de la operación del software con una GUI es cuando se relaciona con un elemento de control. El programa registra el valor de esa acción elegida y realiza los comandos prescritos en el código. Los menús de interfaz con el usuario, los botones, los menús desplegables, los controladores deslizantes y el texto editable son elementos que controlan las operaciones del software. Al realizarse la ejecución de las instrucciones, el control vuelve al GUI para que puedan realizarse la siguiente acción requerida por el usuario. Este ciclo se repite hasta que se cierra la GUI.

Para llegar a entender a un nivel básico la forma de programar una interfaz, solo se necesita entender cinco comandos:

uimenu, uicontrol, get, set y axes.

No obstante, lo que hace verdaderamente refinados a estos comandos es el gran número de maneras de uso. Es difícil describir todos lo tipo de situaciones, pues requiere demasiado espacio y sería muy costoso frente al tiempo.

La interfaz gráfica está creado por dos archivos uno (*.m) y otro (*.fig). Cada vez que se adicione un nuevo elemento en la interfaz gráfica, se genera automáticamente código en el archivo (.m).

En el archivo (*.m), nos encontramos con un procesador de texto donde mencionamos todas las funciones que hacen referencia a todos los objetos presentado en el archivo (*.fig). Las funciones creadas por el comando “function”, hacen referencia al objeto mencionado por la denominación del nombre. Los comandos y funciones, componen el nuevo archivo de texto llamado (M-file) ó (*.m).

(M-file) pueden ser de scripts o funciones. Los scripts son simplemente los archivos que contienen una secuencia de sentencias de Matlab.

El archivo (*.fig), es el archivo que se va ha visualizar como nuestra interfaz gráfica. Aquí pondremos los diversos objetos, botones, editores de texto, panel de control, gráficas, etc…

Todos los valores de las propiedades de los elementos (color, valor, posición, string…) y los valores de las variables transitorias del programa se almacenan en una estructura, los cuales son accedidos mediante un único y mismo identificador para todos éstos.

Ejemplo:

```
[Nombre-variable] = 1;
handles.[Nombre-variablesalida] = [Nombre-variable];
Guidata (hObject, handles);
```

En este caso, vemos que el valor “1” queda grabado y puede ser llamado a través de la variable. El valor queda memorizado gracias al manejador o también llamado identificador. Handles, es nuestro identificador de los datos de la aplicación. Las variables no quedarían memorizadas sin el comando:

```
guidata (hObject, handles);
```

Guidata, es la sentencia para salvar los datos de la aplicación.

Guidata es la función que guarda las variables y propiedades de los elementos en la estructura de datos de la aplicación, por lo tanto, como regla general, en cada subrutina se debe escribir en la última línea lo siguiente:

```
guidata (hObject, handles);
```

Esta sentencia nos garantiza que cualquier cambio o asignación de propiedades o variables quede almacenado.

Por ejemplo, si dentro de una subrutina una operación dio como resultado una variable “coche” para poder utilizarla desde el programa u otra subrutina debemos salvarla de la siguiente manera:

```
handles.coche = rojo;
guidata (hObject, handles);
```

La primera línea crea la variable llamada “coche” a la estructura de datos de la aplicación apuntada por handles y la segunda graba el valor.

3.2. ¿Cómo se arranca un GUI en Matlab?

Inicio

Para iniciar nuestro proyecto, lo podemos hacer de dos maneras:

1. Ejecutando la siguiente instrucción en la ventana de comandos:

   >> `guide`

2. Haciendo un clic en el ícono que muestra la figura 1:

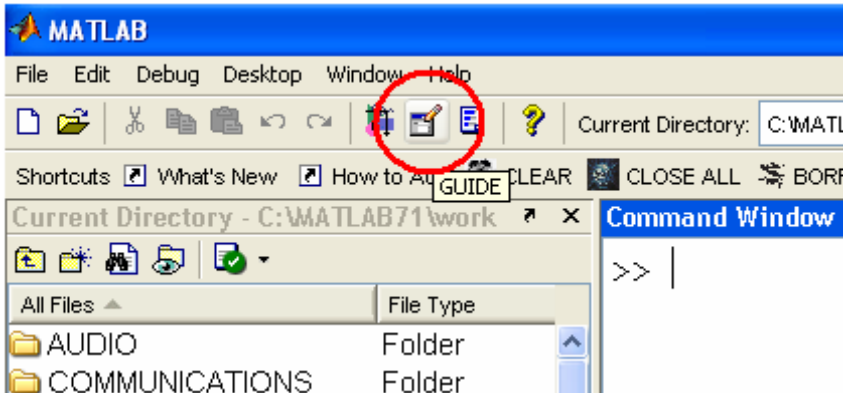

Figura 1. Icono GUI.

Se presenta el siguiente cuadro de diálogo que muestra la figura 2:

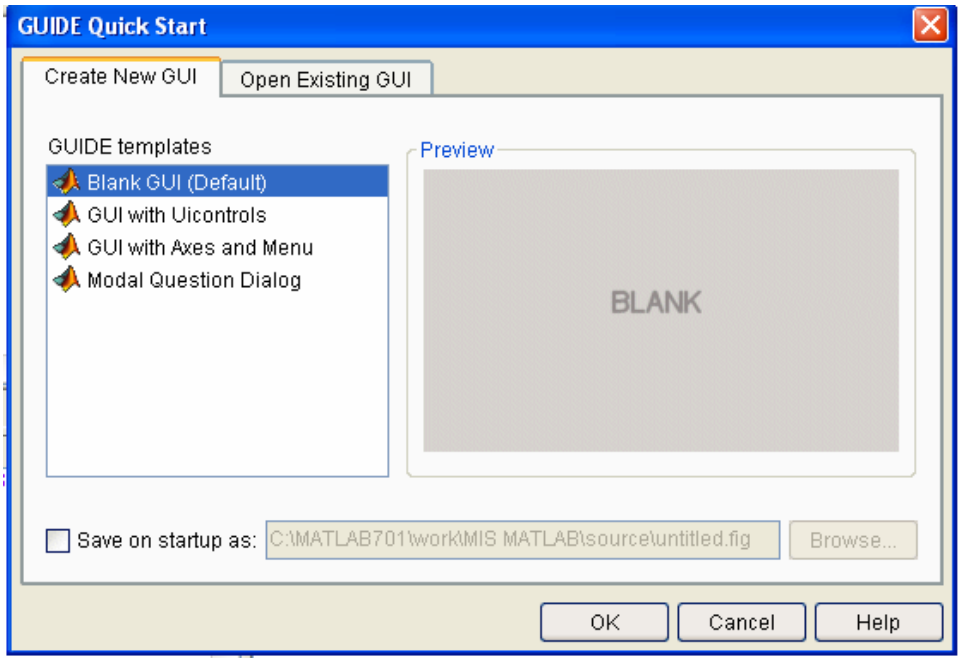

Figura 2. Ventana de inicio GUI.

Se presentan las siguientes opciones:

- Blank GUI (Default)

La opción de interfaz gráfica de usuario en blanco (viene predeterminada), nos presenta un formulario nuevo, en el cual podemos diseñar nuestro programa.

- Gui con Uicontrols

Esta opción presenta un ejemplo en el cual se calcula la masa, dada la densidad y el volumen, en alguno de los dos sistemas de unidades. Podemos ejecutar este ejemplo y obtener resultados.

- Gui con Axes y Menú

Esta opción es otro ejemplo, el cual contiene el menú File con las opciones open, print y close. En el formulario tiene un Popup menú, un push button y un objeto axes, podemos ejecutar el programa eligiendo alguna de las seis opciones que se encuentran en el menú despegable y haciendo clic en el botón de comando.

- Modal Question Dialog

Con esta opción se muestra en la pantalla un cuadro de diálogo común, el cual consta de una pequeña imagen, una etiqueta y dos botones “Yes” y “No”, dependiendo del botón que se presione, el GUI retorna el texto seleccionado.

Elegimos la primera opción, Blank GUI, y tenemos la ventana que muestra la figura 3:

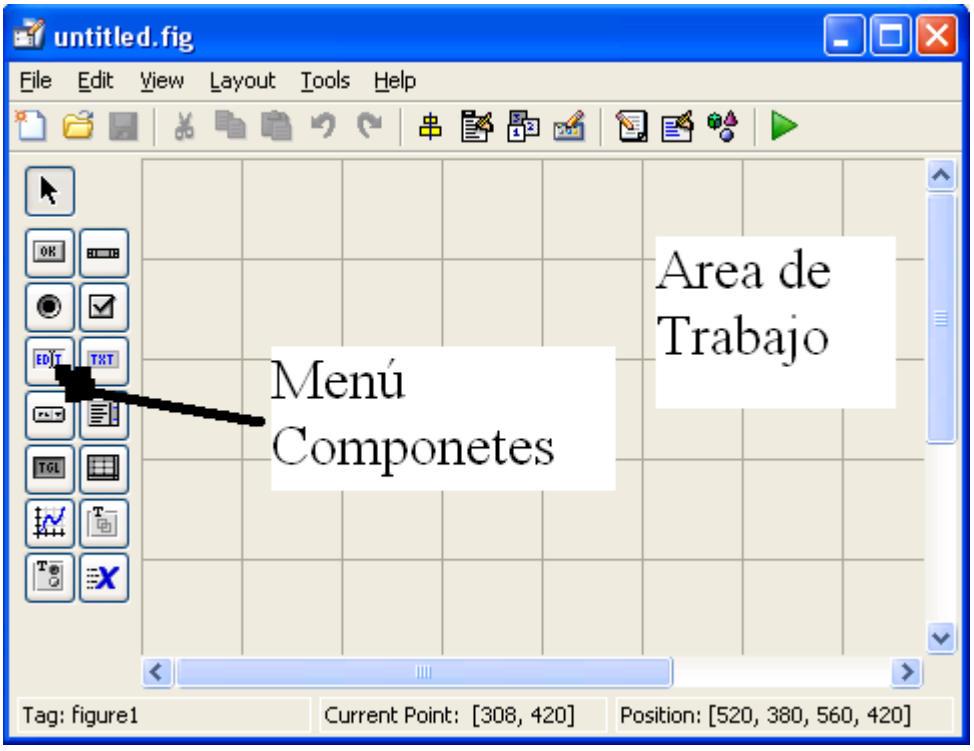

Figura 3. Entorno de diseño GUI

La interfaz gráfica cuenta con las siguientes herramientas:

Alineación de Componentes (Alignment tool): esta opción permite alinear los componentes que se encuentra en el área de trabajo (Layout Área) de manera personalizada.

Editor de menú. Editor de Menús (Menú Editor): El redactor de Menú crea menús de ventana y menús de contexto.

Editor de orden de etiqueta, enumera y ordena los elementos

Editor del M-file, presenta el archivo donde se escribe el código

Propiedades de objetos (Property Inspector): con esta opción se asignan y modifican las propiedades de cada objeto en forma personalizada.

Navegador de objetos. Navegador de Objetos (Object Browser): Muestra todos los objetos que se encuentra en la figura (en forma de árbol) y a través de Object Browser se puede seleccionar los objetos.

Botón de ejecución (Run button). Grabar y ejecutar: Al presionarse crea la figura de la interfaz diseñada en el Área.

CAPÍTULO 4. SIMULACIÓN DINÁMICA DEL ARRANQUE DE UN MOTOR DE INDUCCIÓN

4.1.- MODELO DE LA MÁQUINA DE INDUCCIÓN. [3]

4.1.1. Introducción

Una forma de representar las ecuaciones de las máquinas eléctricas es a través de los fasores espaciales, que nos permiten reducir el número de ecuaciones necesarias y cambiar de sistemas de referencia. El módulo del fasor espacial de una magnitud cualquiera proporciona la amplitud de la onda espacial y su posición en el plano complejo da la posición espacial del máximo de dicha onda.

En este capítulo se definen los fasores espaciales de las variables del motor de inducción en el marco de referencia del estátor [dq]. Las ecuaciones dinámicas del motor de inducción se expresan en este marco. La idea de un marco de referencia giratorio [DQ], se introduce para transformar los componentes de corriente alterna de los vectores del estátor en señales de corriente continua. Terminamos explicando la adaptación de las ecuaciones dinámicas del motor de un marco de referencia giratorio.

Los fasores de la tensión del estátor en el espacio y los flujos magnéticos (concatenaciones de flujo) se emplean comúnmente en el análisis y control de motores de inducción.

4.1.2 Representación fasorial de una máquina de inducción de estátor trifásico

El fasor espacial de la fuerza magnetomotriz resultante en el estátor de un sistema trifásico simétrico, cuyas fases están desplazadas entre sí un ángulo $2\pi/3$, cuyo número de vueltas de cada fase son iguales y por las que circulan las corrientes $i_{sA}(t)$, $i_{sB}(t)$, $i_{sC}(t)$, es :

$$\bar{f}_s = \bar{f}_{sA} + \bar{f}_{sB} + \bar{f}_{sC} \tag{4.0}$$

donde

$$\bar{f}_{sA} = \frac{2}{\pi p} N_s k_s i_{sA}(t) e^{j0} \tag{4.1}$$

$$\bar{f}_{sB} = \frac{2}{\pi p} N_s k_s i_{sB}(t) e^{j2\pi/3} \tag{4.2}$$

$$\bar{f}_{sC} = \frac{2}{\pi p} N_s k_s i_{sC}(t) e^{j4\pi/3} \tag{4.3}$$

Si sustituimos $a=e^{j2\pi/3}$, tenemos:

$$\bar{f}_s = \frac{2}{\pi p} N_s k_s \left[i_{sA}(t) + a i_{sB}(t) + a^2 i_{sC}(t) \right] = \frac{2}{\pi p} N_s k_s \bar{i}_s \qquad (4.4)$$

Donde I_s es el fasor de las corrientes del estátor en el sistema de referencia estacionario.

De donde podemos sacar la conclusión de fasor espacial de una magnitud cualquiera que esté distribuida sinusoidalmente en el espacio, de forma que para una magnitud denominada por x, su fasor espacial será

$$\bar{x}_k = \frac{1}{c} \left[x_{kA}(t) + a x_{kB}(t) + a^2 x_{kC}(t) \right] \qquad (4.5)$$

Donde, $k=s$ en caso de que se trate de una magnitud del estátor y $k=r$ si se trata de una magnitud del rotor.

Ecuación del estátor:

$$X_s^s = [Xa\,(t) + X_b\,(t) \cdot e^{j2p/3} + X_c\,(t) \cdot e^{j4p/3}] = [\,X_d + jX_q\,] \qquad (4.6)$$

X_s^s Vector estátor total

X_a (t) Vector de la fase a

X_b (t) Vector de la fase b

X_c (t) Vector de la fase c

X_d Vector (d) en los ejes cartesianos (dq)

X_q Vector (q) en los ejes cartesianos (dq)

Las magnitudes del rotor giran entorno al rotor a frecuencia de deslizamiento y el rotor gira entorno al estátor a w

Transformación del rotor al estátor.

$$X_r^s = X_r^r \cdot e^{jwt} \tag{4.7}$$

X_r^s Vector del rotor expresado en el sistema de referencia del estátor

X_r^r Vector del rotor expresado en el sistema de referencia del rotor

Con esto, podemos definir el fasor de la tensión del estátor,

$$\overline{U}_s^{(s)} = \frac{2}{3}\left[u_{sA}(t) + a u_{sB}(t) + a^2 u_{sC}(t)\right] \tag{4.8}$$

$$\bar{I}_s^{(s)} = \frac{2}{3}\left[i_{sA}(t) + a i_{sB}(t) + a^2 i_{sC}(t)\right] \tag{4.9}$$

donde el superíndice *(s)* indica el sistema de referencia en el que la ecuación está expresada.

Tenemos el mismo resultado para la corriente en el rotor, teniendo en cuenta que está compuesta por, $i_{ra}(t)$, $i_{rb}(t)$ e $i_{rc}(t)$ que son los valores instantáneos de las corrientes del rotor a través de las fases *a*, *b* y *c*. Y para la tensión en el rotor se procedería de la misma manera, pero al considerar rotores en cortocircuito, la tensión es cero.

Las expresiónes para los fasores del flujo en el estátor y en el rotor son:

$$\overline{\Psi}_s^{(s)} = \frac{2}{3}\left[\psi_{sA}(t) + a\psi_{sB}(t) + a^2\psi_{sC}(t)\right] \tag{4.10}$$

$$\overline{\Psi}_r^{(r)} = \frac{2}{3}\left[\psi_{ra}(t) + a\psi_{rb}(t) + a^2\psi_{rc}(t)\right] \tag{4.11}$$

Las cuales, forman parte del grupo de ecuaciones que modelan la máquina asíncrona.

4.1.3 Ecuaciones diferenciales que describen el comportamiento de la máquina asíncrona según las expresiones matemáticas de la teoría de los fasores espaciales.

Ecuación de balance de tensiones en el estátor en el sistema de de referencia ligado al estátor:

$$\overline{U}_s^{(s)} = R_s \bar{I}_s^{(s)} + \frac{d\overline{\Psi}_s^{(s)}}{dt} \tag{4.12}$$

R_s Es la resistencia de un devanado del estátor.

Ecuación de balance de tensiones del rotor en el sistema de referencia ligado al rotor:

$$0 = R_r \bar{I}_r^{(r)} + \frac{d\overline{\Psi}_r^{(r)}}{dt} \tag{4.13}$$

Las ecuaciones (4.10) y (4.11) en el sistema general de coordenadas,

$$\overline{U}_s^{(g)} = R_s \bar{I}_s^{(g)} + \frac{d\overline{\Psi}_s^{(g)}}{dt} + j\omega_g \overline{\Psi}_s^{(g)} \tag{4.14}$$

$$0 = R_r I_r^{(g)} + \frac{d\overline{\Psi}_r^{(g)}}{dt} + j\left(\omega_g - \omega_r\right)\overline{\Psi}_r^{(g)} \tag{4.15}$$

a partir de las cuales se pueden particularizar para cualquier sistema de referencia deseado.

4.1.4 Ecuación del par en el sistema de referencia general

$$M_i = -p\frac{3}{2}\mathrm{Im}\left(\overline{\Psi}_s^{(g)} \bar{I}_s^{(g)^*}\right) = p\frac{3}{2}\mathrm{Im}\left(\overline{\Psi}_r^{(g)} \bar{I}_r^{(g)^*}\right) \tag{4.16}$$

La cual, forma parte del grupo de ecuaciones que modelan la máquina asíncrona.

4.1.5 Circuito equivalente

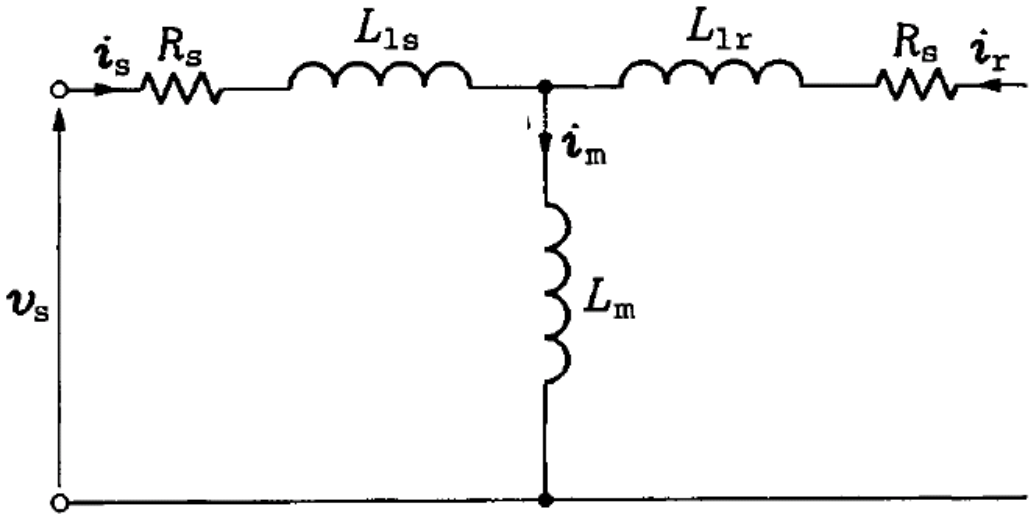

Figura 5. Circuito Equivalente.

4.2.- MÉTODOS DE ARRANQUE SIMULADOS.

El archivo Simulink “comparaciónUf” nos simula los métodos de arranque en un motor de inducción. En este caso utilizaremos tres métodos:

- Arranque directo, con la amplitud y la frecuencia constante.
- Arranque con la amplitud y la frecuencia variables.
- Arranque con la amplitud fija y la frecuencia variable.

Archivo Simulink “comparaciónUf” que muestra la figura 7:

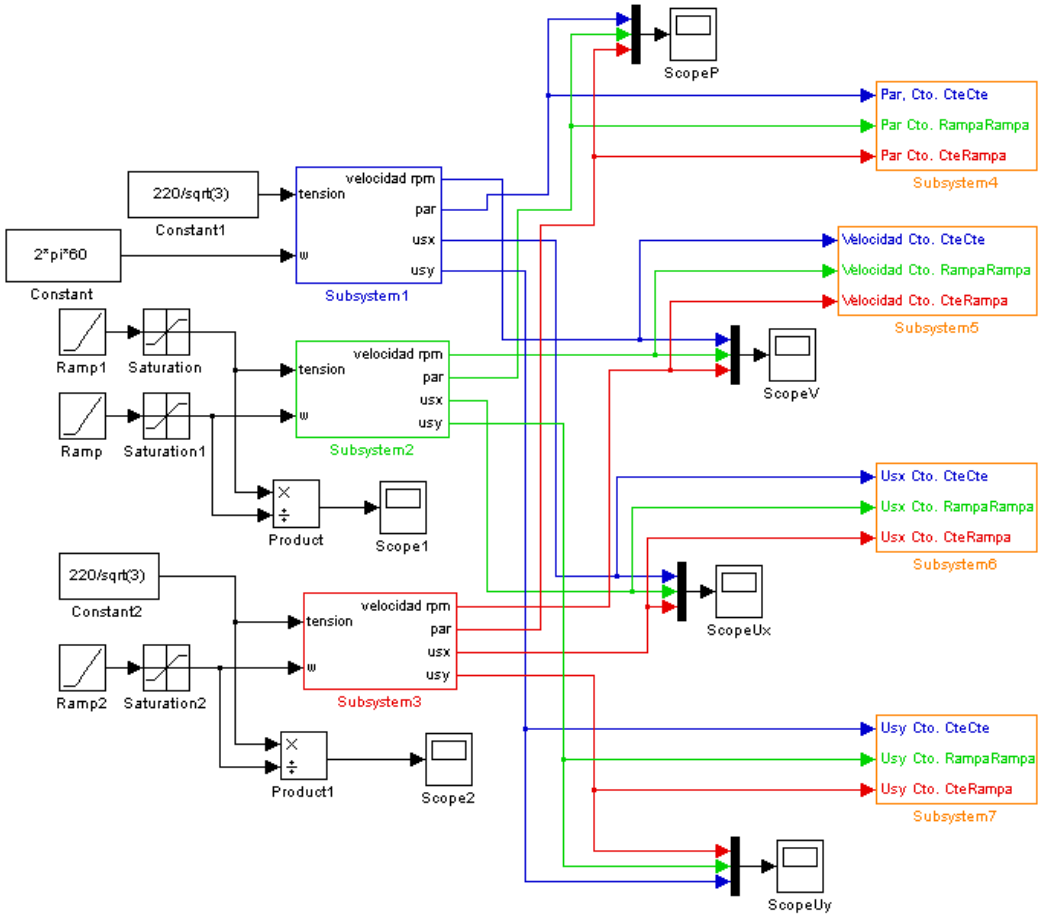

Figura 7. Archivo Simulink “comparaciónUf”

El primer bloque (Subsystem1) hace referencia al primer arranque, figura 8:

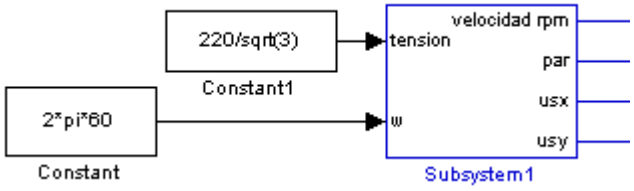

Figura 8. Arranque directo.

Este subsistema contienen a su vez otros dos subsistemas, figura 9:

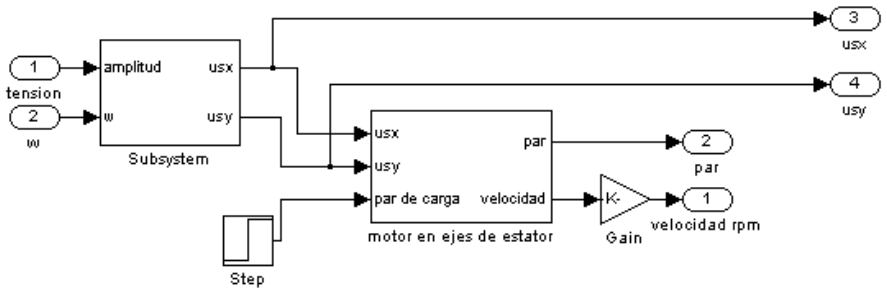

Figura 9. Subsistema1.

Donde el primero calcula la tensión y el segundo calcula el par y la velocidad. Si abrimos el segundo subsistema, podremos hallar en la figura 10:

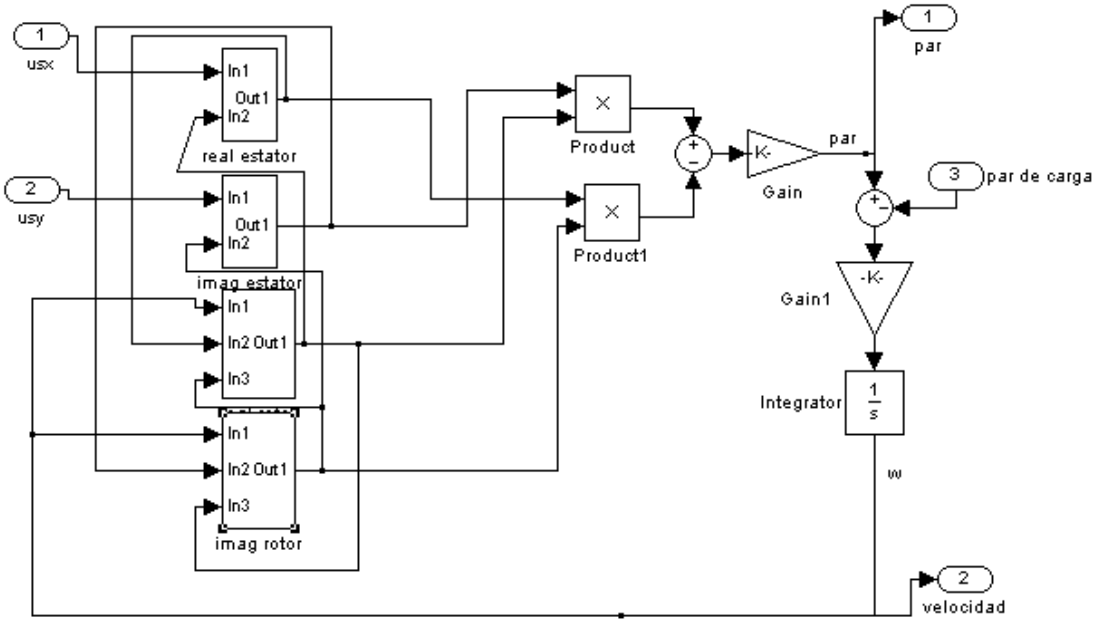

Figura 10. Subsistema.

Donde vemos el cálculo del par de carga y de la velocidad en la parte de la derecha. Si abrimos el bloque “imag rotor” en la figura 11:

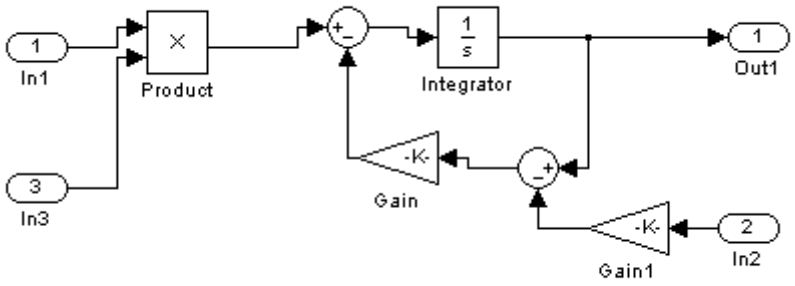

Figura 11. Subsistema.

CAPÍTULO 5. INTERFAZ GRÁFICO PARA EL USUARIO.

5.1 INTRODUCCIÓN

Explicación de la interfaz.

La finalidad de esta interfaz es poder controlar archivos Simulink. El control se tiene sobre los bloques que forma el circuito de cualquier sistema Simulink. Estos bloques contienen parámetros cuyos valores pueden ser modificados por nuestro GUI.

La ventana de trabajo Simulink, tiene parámetros cuyos valores también pueden ser modificados por nuestro GUI.

Nos centraremos en el archivo Simulink llamado “ComparaciónUf”, es un circuito que simula los arranques de un motor de inducción. El archivo Simulink está formado por las ecuaciones diferenciales que denominan a una máquina eléctrica.

Está compuesto por un archivo GUI llamado PFCSGM, cuya imagen muestra la figura 12. Es una ventana de trabajo en la cual nos encontramos con los mandos de control de un interfaz grafica.

5.2 PANEL DE CONTROL GENERAL

En este apartado explicaremos, punto por punto, todos los elementos de nuestro GUI. Presentamos la imagen principal de nuestro GUI en la figura 12:

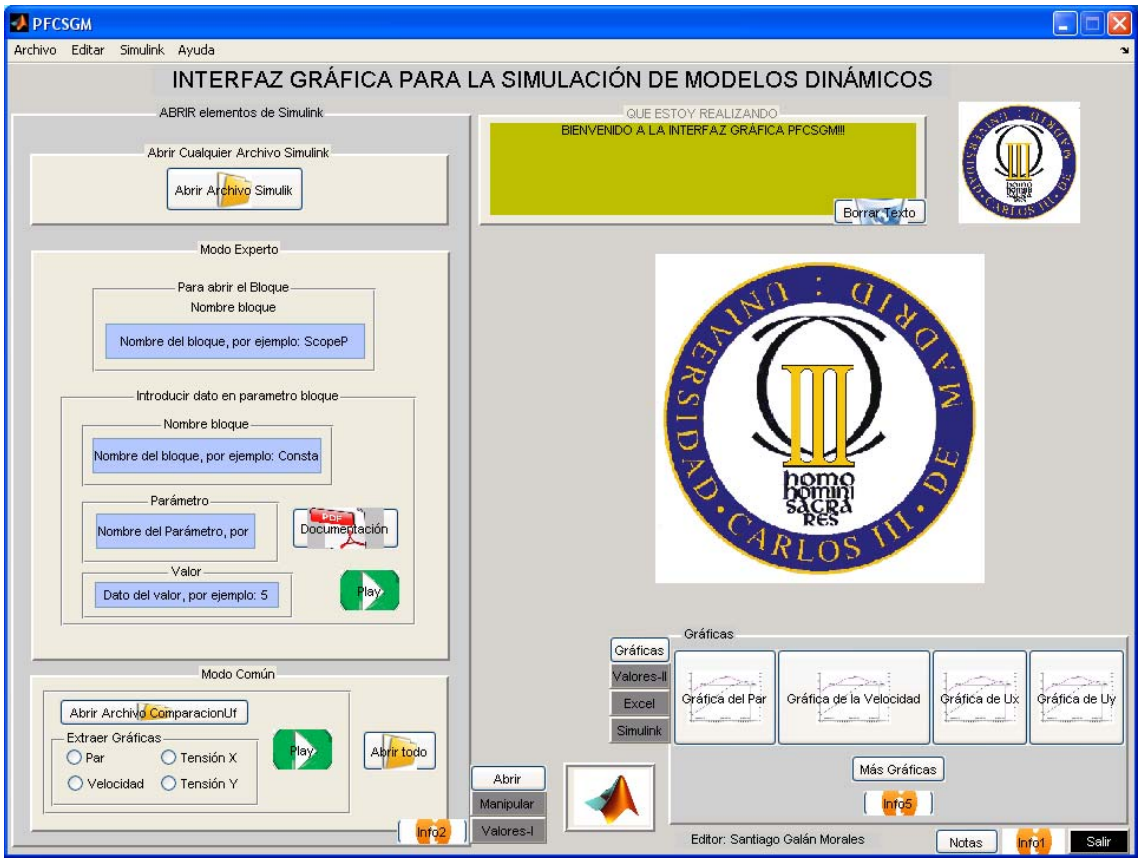

Figura 12. GUI.

En él, podemos encontrarnos varios elementos, que en adelante explicaremos punto por punto.

5.2.1 Menú herramientas:

En primer lugar nos encontramos un menú herramientas en la parte superior de nuestro GUI como muestra la figura 13:

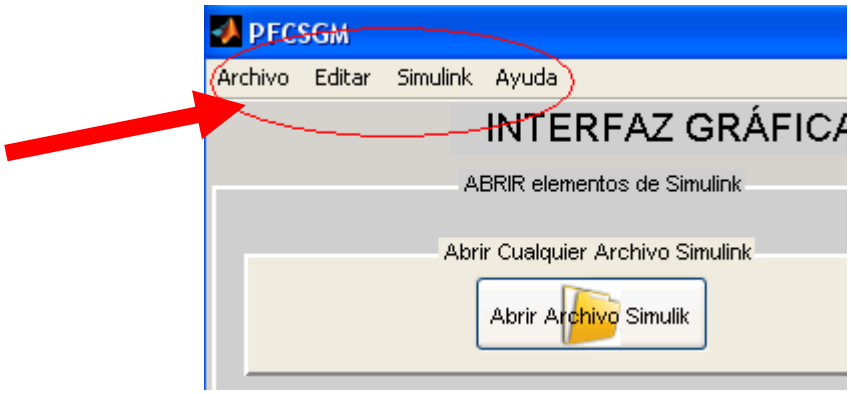

Figura 13. Menú herramientas.

En él podemos optar a las acciones más típicas que ofrece Windows donde nos encontramos dentro de cada acción con varias posibles selecciones (distingamos acción de selección):

En la primera acción de menú, nos encontramos:

- Archivo
  - Abrir
    - Abrir archivos *.fig
    - Abrir archivos *.m
    - Abrir archivos *.mdl
    - Abrir archivos *.doc
    - Abrir archivos *.pdf
    - Abrir archivos *.xls
  - Imprimir
  - Salir

Con la primera selección, llamada “abrir”, podemos abrir archivos (*.fig), que son archivos GUI ejecutables, también podemos abrir archivos (*.m), que son archivos de programación, podemos abrir archivos (*.mdl), que son sistemas Simulink, podemos abrir archivos (*.doc), que son archivos de redacción y lectura, podemos abrir archivos (*.pdf), que son archivos lectura y podemos abrir archivos (*.xls), que son archivos contables. Gracias a una ventana emergente, en cada una de las opciones, que nos muestra un directorio.

La segunda selección es la de “imprimir”, con esto, podemos encontrar impresoras que nos documente en papel las imágenes de nuestro GUI.

Y por tercera selección, tenemos “salir”, que nos permite cerrar la interfaz así como el archivo Simulink abierto.

En la segunda acción de menú, nos encontramos:

- Editar
  - Aumentar tamaño de texto
  - Disminuir tamaño de texto

Con la primera selección del menú “editar”, nos encontramos con dos selecciones:

- Aumenta el tamaño de texto, que nos aumenta el tamaño de texto del panel de control de texto general.
- Disminuye el tamaño de texto, que nos aumenta el tamaño de texto del panel de control de texto general.

En la tercera acción de menú, nos encontramos:

- Simulink
  - Abrir archivo Simulink

Con esta selección del menú “Simulink”, nos permite abrir cualquier archivo Simulink, gracias a una ventana emergente, que nos muestra un directorio.

En la cuarta acción tenemos “ayuda”:

- Ayuda
  - Ayuda en la Web
  - Ayuda desde archivo

Ayuda en la Web, nos abre una página Web donde podemos recabar información sobre todo lo relacionado con Matlab.

Ayuda desde archivo de ayuda, es una acción que abre un archivo editado con información sobre nuestra interfaz gráfica.

Antes de programar tenemos que crear las funciones de cada acción y de cada selección. Las funciones se crean cuando creamos las distintas acciones y selecciones en el menú editor de la ventana (*.fig) que muestra la figura 14:

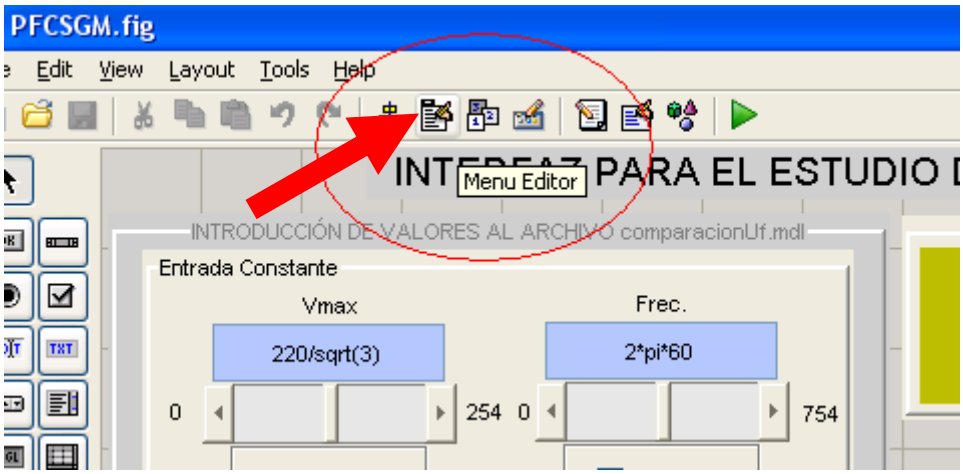

Figura 14. Menú editor.

Una vez pulsado el menú editor, nos saldrá una ventana emergente como muestra la figura 15:

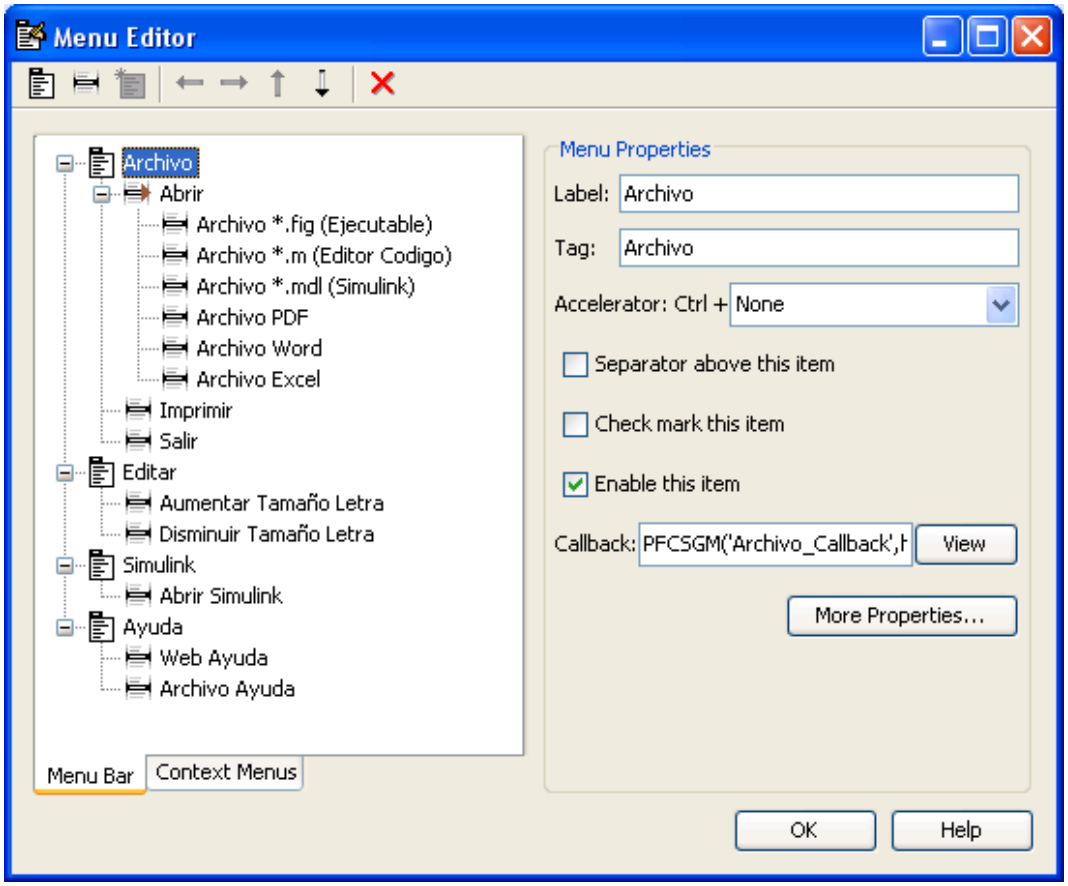

Figura 15. Menú editor.

Como vemos, aquí es donde se crea las acciones y selecciones. Una vez completado "el árbol" Matlab crea las funciones de cada una de las acciones y de cada una de las selecciones en la ventana (*.m) que es la ventana editora de código. En adelante explicaremos la programación de estas acciones y selecciones aunque omitiremos las que sean iguales en su programación.

La figura 16 muestra la imagen de la acción "Archivo".

Figura 16. Acción "Archivo".

Cuya programación es:

Los comentarios que precede a la programación dentro del propio programa se indican con el signo (%) y de un color verde. Cuya programación será de color violeta.

```
%Función Archivo: No ejecuta ninguna acción, sirve de ruta.
function Archivo_Callback (~, ~, handles)

     %Introduce un mensaje de texto en el panel de texto
     general.
          set (handles.text,'string','Archivo...');

     %Guarda el nombre de la función
          guidata (Archivo_Callback);
```

La figura 17 muestra la imagen de la acción "Archivo fig".

Figura 17. Selección Archivo *.fig

La programación de la selección "Archivo (*.fig)" es:

```
%Se define la función archivofig, abre los archivos
(*.fig). Archivofig_Callback es el nombre de la función.
     function Archivofig_Callback (~, ~, handles)

     %Introduce un mensaje de texto en el panel de texto
     general, handles.text: especifica el objeto; 'string':
     especifica el tipo de dato, 'elige un archivo
     (*.fig)': especifica el texto
          Set(handles.text,'string', 'Elige un archivo
          (*.fig)');
```

```
%Abre una ventana emergente para seleccionar un
archivo GUI y cargarlo. FileName, Path: Son simples
variables. En FileName se guarda el nombre del archivo
y en Path la ruta. {'*.fig', 'Archivo GUI
ejecutable…(*.fig)': Impone la extensión del archivo a
abrir. 'Abrir Archivo GUI ejecutable...': Pone un
texto de información. 'PFCSGM.fig': Pone el nombre del
archivo por defecto. 'MultiSelect', 'on': Parámetro y
valor, permite la selección de varios archivos.
     [FileName Path] = uigetfile ({'*.fig', 'Archivo
     GUI ejecutable...(*.fig)'}, 'Abrir Archivo GUI
     ejecutable...', 'PFCSGM.fig', 'MultiSelect',
     'on');

%Condición de retorno en el caso de que le demos a
cancelar
     if isequal (FileName,0)
          %Retorna al GUI
          return
     else
          %Abre el archivo seleccionado
          winopen (strcat (Path,FileName));
     end

%Introduce un mensaje de texto en el panel de texto
general, FileName: nombra el contenido de esa variable
     Set (handles.text, 'string', ['Ha elegido abrir
     el archivo:', FileName])

%Guarda el nombre de la función
     guidata (Archivofig_Callback);
```

La figura 18 muestra la imagen de la acción "Imprimir".

Figura 18. Selección Imprimir.

La programación de la selección "Imprimir" es:

```
%Se define la función Imprimir
```

```
function Imprimir_Callback (~, ~, handles)

%Introduce un mensaje de texto en el panel de texto
general
     set (handles.text, 'string', 'Has elegido la
     opción de imprimir');

%Imprime la ventana de dialogo, nuestro GUI
     printdlg (handles.figure1)

%Guarda el nombre de la función
     guidata (Imprimir_Callback);
```

La figura 19 muestra la imagen de la acción "Salir".

Figura 19. Selección "Salir"

Esta selección nos permite cerrar la interfaz gráfica, así como cerrar el archivo Simulink abierto en el proceso. Al pulsar la selección, saldrá una ventana emergente que nos preguntará si de verdad queremos salir de la interfaz como muestra la figura 20:

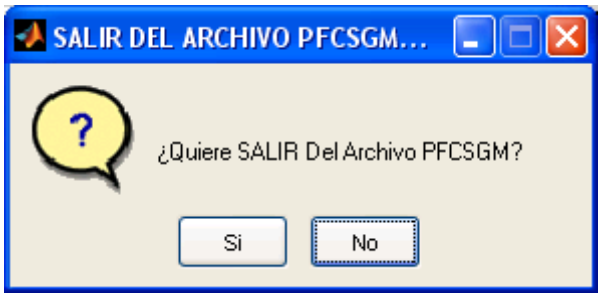

Figura 20. Ventana emergente, opciones.

En el caso de pulsar "no", retornaremos a la interfaz y si pulsamos "si", cerrará el archivo Simulink abierto, cerrará la interfaz y mostrará un mensaje de despedida en una ventana emergente como muestra la figura 21:

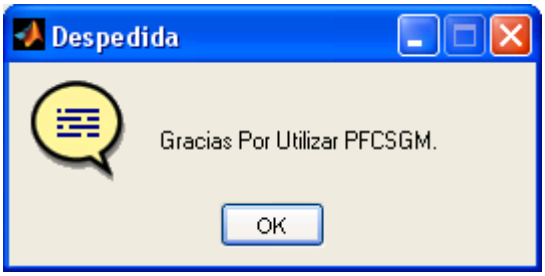

Figura 21. Ventana emergente, despedida.

La programación de la selección “Salir” es:

```
%Se define la función “Salir2”: Cierra nuestro GUI y cierra
el archivo Simulink abierto.
     function Salir2_Callback (~, ~, handles)

     %Introduce un mensaje de texto en el panel de texto
     general
          Set (handles.text, ‘string’, ‘Has elegido la
          opción de salir’);

     %Ventana emergente que propone dos o más soluciones
          Selection = questdlg ([‘¿Quiere SALIR Del
          Archivo’ get(handles.figure1, ‘name’) ‘?’ ],
          [‘SALIR DEL ARCHIVO’ get(handles.figure1, ‘Name’)
          ‘...’], ‘Si’, ‘No’, ‘No’);

     %Si eliges cancelar, retorna al GUI
     if strcmp (selection, ‘No’)

          %Retorna a nuestro GUI
          return
     end

     %Si eliges "si" sigue con la función y se encuentra
     con la salida del GUI
          delete (gcf)


     %Mensaje de despedida en una ventana emergente
          Msgbox (‘Gracias Por Utilizar PFCSGM.’,
          ‘Despedida’, ‘help’)

     %Restablece los valores iniciales en el caso de que
     sea el archivo Simulink “compacionUf”
     %Compara si el archivo abierto es “ComparaciónUf”
     %Comparación de palabras
          cond = strcmp (‘“ComparaciónUf”’, handles.ruta);
```

```
%Si está abierto, introducirá los valores iniciales en
tal archivo
if cond == 1
     set_param('ComparaciónUf/Constant1', 'value',
     '220/sqrt(3)');
     set_param('ComparaciónUf/Constant', 'value',
     '2*pi*60');
     set_param('ComparaciónUf/Ramp1', 'slope',
     '220/sqrt(3)-10)/0.75');
     set_param('ComparaciónUf/Ramp', 'slope',
     '120*pi/0.75');
     set_param('ComparaciónUf/Constant2', 'value',
     '220/sqrt(3)');
     set_param('ComparaciónUf/Ramp2', 'slope',
     '120*pi/.75');
end

%Restablece los valores generales de la ventana de
trabajo de Simulink
     set_param(handles.ruta, 'StopTime', '1.8');
     set_param(handles.ruta, 'StartTime', '0.0');
     set_param(handles.ruta, 'Solver', 'ode45');

%Salvar archivo Simulink abierto
     save_system (handles.FileName);

%Cerrar archivo Simulink abierto
     close_system (handles.FileName);

%Guarda el nombre de la función
     guidata (Salir2_Callback);
```

La figura 22 muestra la imagen de la acción “editar”.

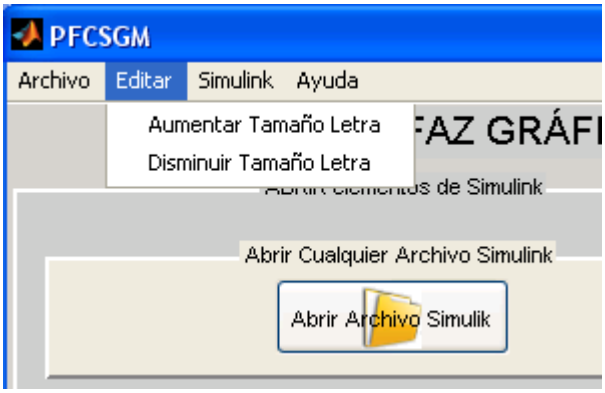

Figura 22. Selección Aumentar Tamaño Letra.

La programación de la selección “Aumentar Tamaño Letra” es:

```
%Función aumentarletra: nos permite aumentar la letra de
nuestro panel de texto general
     function aumentarletra_Callback (~, ~, handles)

     %Introduce un mensaje de texto en el panel de texto
     general
          Set (handles.text, ‘string’, ‘Has elegido la
          opción de aumentar el tamaño de letra’);

     %Devuelve la propiedad del tamaño de texto del panel
     de texto general
          currentFontSize = get(handles.text, ‘FontSize’);

     %Aumenta el valor del tamaño de texto en una unidad.
          set(handles.text, ‘FontSize’, currentFontSize+1);

     %Guarda el nombre de la función
          guidata (aumentarletra_Callback);
```

La figura 23 muestra la imagen de la acción “web ayuda”.

Figura 23. Selección “Web Ayuda”.

La programación de la selección “Web Ayuda” es:

```
%Función Ayuda: nos da información sobre nuestro GUI
     function WebAyuda_Callback (~, ~, handles)

     %Introduce un mensaje de texto en el panel de texto
     general
          set(handles.text,‘string’,‘Se abrirá una página
          Web donde podrá recabar información’);

     %Abre una pagina Web donde podemos recabar información
     que nos sirva de más ayuda
          Web http://www.mathworks.es/index.html

     %Guarda el nombre de la función
          Guidata (WebAyuda_Callback);
```

La figura 24muestra la imagen de la acción “archivo ayuda”.

Figura 24. Selección de “Archivo Ayuda”.

La programación de la selección “Archivo Ayuda” es:

```
%Función Ayuda: nos da información sobre nuestro GUI
     function ArchivoAyuda_Callback (~, ~, handles)

     %Introduce un mensaje de texto en el panel de texto
     general
          set (handles.text, ‘string’, ‘Se abrirá un
          documento de ayuda’);

     %Abre el documento de ayuda
          open (‘ayuda/ayuda.pdf’)

     %Guarda el nombre de la función
          Guidata (ArchivoAyuda_Callback);
```

5.2.2 Panel de control de texto

Si nos fijamos en la parte derecha superior de nuestra interfaz, veremos un panel de control de texto cuya ilustración muestra la figura25.

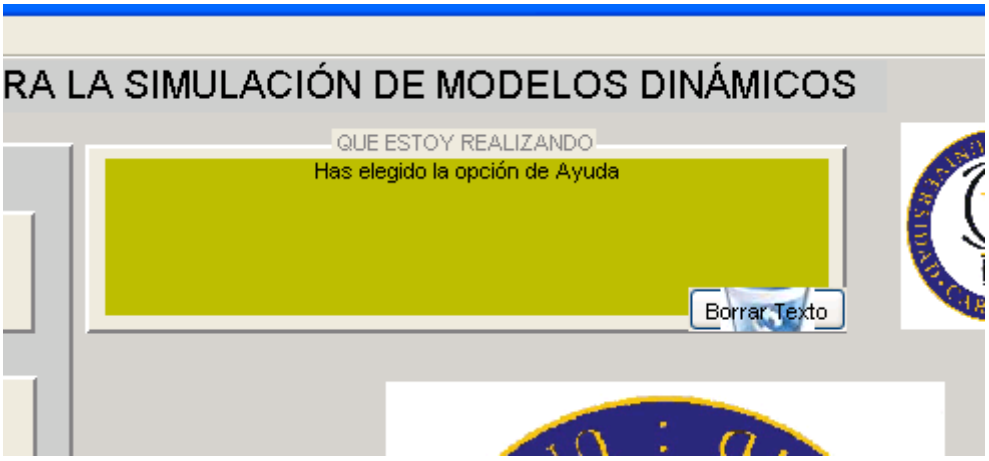

Figura 25. Panel de control de texto

En él, podemos encontrar las indicaciones de cada acción que hagamos con nuestro GUI. Su fin es dar la información de cada paso que vayamos haciendo. Por ejemplo: si pulsamos el botón (pestaña) “Excel” cuya imagen muestra la fugura 26, veremos en el panel de texto un aviso de que está usted trabajando con el panel de control Excel, como muestra la figura 27:

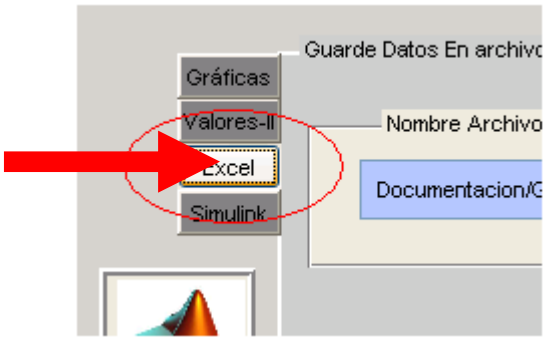

Figura 26. Pestañas

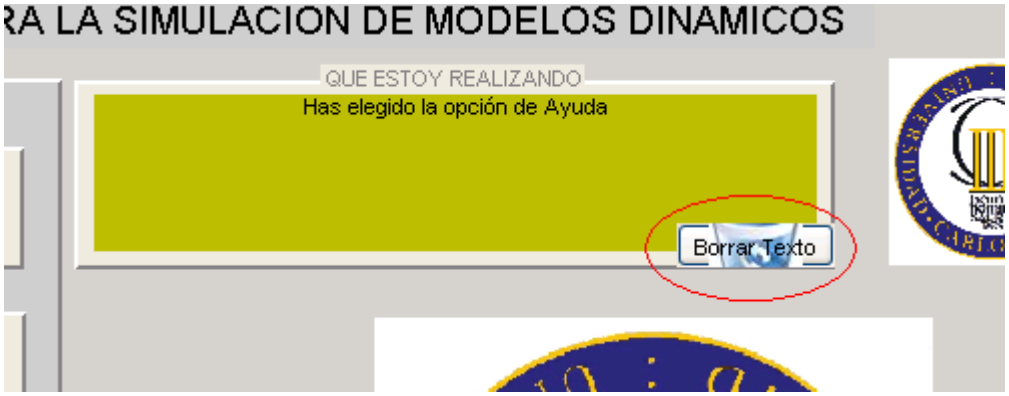

Figura 27. Botón borrar

Este elemento no tiene ningún tipo de de programación, ni tan siquiera tiene su propia función. Tan solo tiene su manejador, handles. [Nombre], al cual se le hace referencia cada vez que queremos introducir un texto en su panel. Si nos fijamos en la parte inferior dcha. del panel, como muestra la figura 27, vemos que existe un botón para eliminar el texto.

La programación del botón borrar es:

```
%Se determina la función "Borrar": este botón nos permite
borrar el texto del panel de texto general
     function BotonBorrar_Callback (~, ~, handles)

     %Muestra un mensaje en el panel de texto general, en
     este caso será un espacio en blanco para que no
     aparezca nada
          Set (handles.text, 'string', ' ');

     %Guarda el nombre de la función
          guidata (BotonBorrar_Callback);
```

5.2.3 Panel gráfico

En la parte central de nuestro GUI, vemos un panel gráfico, que en espera de graficar alguna señal, expone el emblema de la universidad, como muestra la figura 28:

Figura 28. Panel Gráfico

Este panel nos ayuda a mostrar las diferentes señales de onda que nos pueden ofrecer los archivos abiertos de Simulink. Si por ejemplo pulsamos el botón “Gráfica del par” mostrado en la figura 29:

Figura 29. Botón Par

Y si quitamos la selección de la señal de Cto. Cte. Rampa del panel de control “Leyenda (Par)”, mostrada en la figura 30.

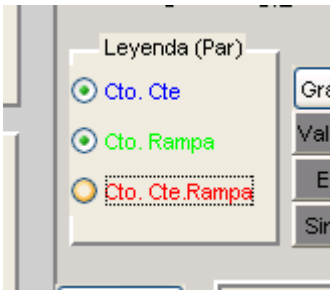

Figura 30, Panel de Control “Leyenda (Par)”.

Obtenemos la señal mostrada en la figura 31:

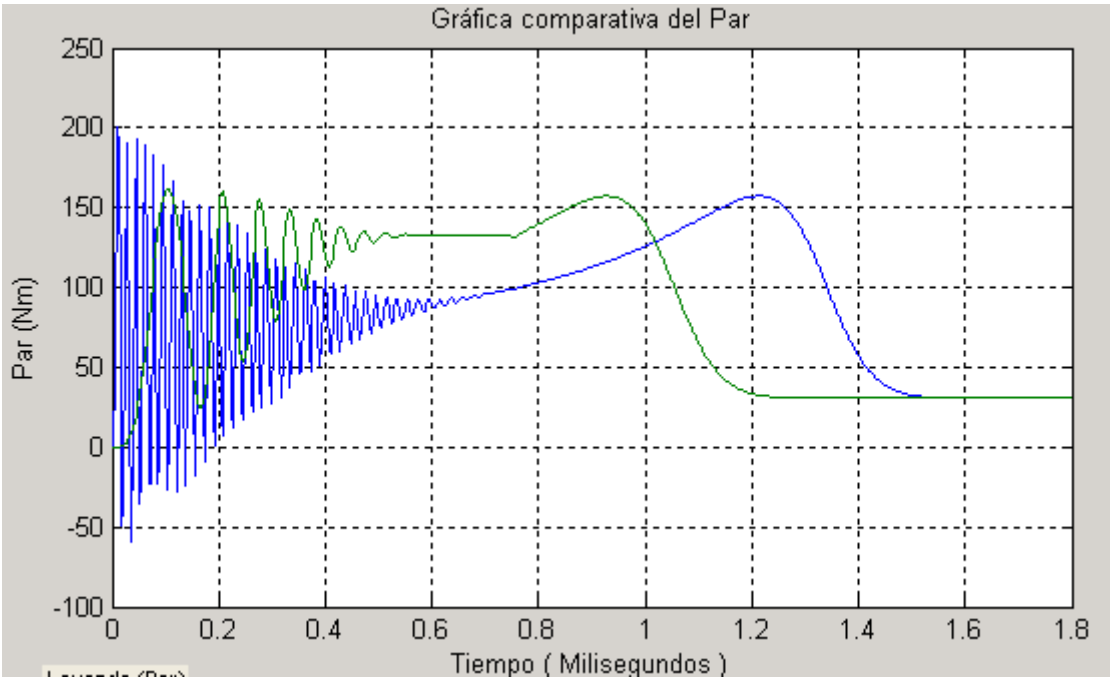

Figura 31. Gráfica del Par

No tiene programación, tampoco hace falta nombrar su función. Tan solo hay que nombrar su manejador (handles. [nombre]) para hacerle referencia.

5.2.4 Botonera del panel general

En la parte inferior derecha, nos encontramos con tres botones, mostrados en la figura 32:

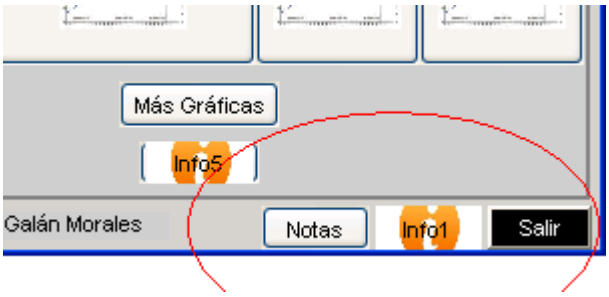

Figura 32. Botonera del panel general

Estos tres botones son:

- Salir
- Info
- Notas

Botón “Salir”:

Este botón hace lo mismo que la selección del menú herramientas.

Su programación:

```
%Función que llama al Botón "Salir" del panel de control
general, nos permite cerrar el GUI y el archivo Simulink
abierto.
     function salir_Callback (~, ~, handles)

     %Introduce un mensaje de texto en el panel de texto
     general
          set(handles.text, ‘string’, ‘Has elegido la
          opción de salir’);

     %Ventana emergente que propone dos o más soluciones
          Selection = questdlg (['¿Quiere SALIR Del
          Archivo’ get(handles.figure1, ‘Name’) ‘?’],
          [‘SALIR DEL ARCHIVO’ get(handles.figure1, ‘Name’)
          ‘...’], ‘Si’, ‘No’, ‘No’);
```

```
%get (handles.figure1, 'name') nos da el nombre de
nuestro GUI
%Condición de cancelar selección
     if strcmp (selection, 'No')
          %Si selecciona "cancelar" retorna al GUI
          return
     end
%Si eliges "si" sigue con la función y se encuentra
con la salida del GUI
     delete (gcf)

%Mensaje de despedida en una ventana emergente
     Msgbox    ('Gracias    Por    Utilizar    PFCSGM',
     'Despedida', 'help')

%Restablece los valores iniciales en el caso de que
sea el archivo Simulink "comapacionUf"
%Comparación de palabras
     cond=strcmp('"ComparaciónUf"',handles.ruta);

%Restablece los valores iniciales al archivo Simulink
"comparaciónUf"
if cond==1
     set_param   ('ComparaciónUf/Constant1',   'value',
     '220/sqrt(3)');
     set_param   ('ComparaciónUf/Constant',   'value',
     '2*pi*60');
     set_param    ('ComparaciónUf/Ramp1',    'slope',
     '(220/sqrt(3)-10)/0.75');
     set_param     ('ComparaciónUf/Ramp',     'slope',
     '120*pi/0.75');
     set_param   ('ComparaciónUf/Constant2',   'value',
     '220/sqrt(3)');
     set_param    ('ComparaciónUf/Ramp2',    'slope',
     '120*pi/0.75');
end

%Restablece los valores generales de la ventana de
trabajo de Simulink
     set_param (handles.ruta, 'StopTime', '1.8');
     set_param (handles.ruta, 'StartTime', '0.0');
     set_param (handles.ruta, 'Solver', 'ode45');

%Salva archivo Simulink
     save_system (handles.FileName);

%Cierra el archivo Simulink abierto
     close_system (handles.FileName);

%Guarda el nombre de la función
guidata (salir_Callback);
```

Botón “Info”:

Este botón nos da información general de nuestra interfaz. Nos indica lo que nos encontramos en ella, tanto los paneles de control, como los diferentes botones. El texto informativo, aparece en el panel de texto general.

Para este primer botón de información, sale una ventana emergente, la cual nos explicará brevemente, en que consiste un botón de información. Esta ventana emergente, solo aparecerá en este botón de información.

Esta ventana emergente se muestra en la figura 34:

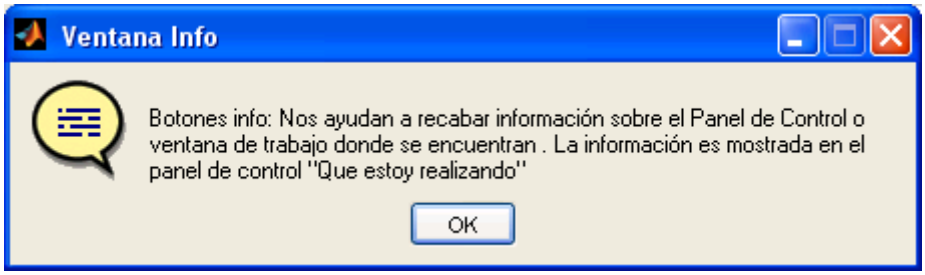

Figura 34. Ventana de información.

Su programación es:

```
%Función que llama al botón "info1" del panel de control
general, nos muestra en el panel de texto general la
información necesaria para poder dar nuestros primeros
pasos dentro de nuestro GUI
function info1_Callback (~, ~, handles)

    %Muestra un mensaje en el panel de texto general
    Set (handles.text, ‘String’, ‘En esta interfaz, nos
    encontramos dos paneles de control gobernados con sus
    respectivas pestañas. También nos encontramos un panel
    informativo llamado "que estoy realizando", un panel
    de ejes que nos mostrará las gráficas y por último
    también nos encontramos ciertos botones de utilidad,
    así como un menú de herramientas.’)

    %Muestra una ventana emergente donde se visualiza un
    texto
    Helpdlg (‘Botones info: Nos ayudan a recabar
    información sobre el Panel de Control o ventana de
    trabajo donde se encuentran. La información es
    mostrada en el panel de control "Que estoy
    realizando"’, ‘Ventana Info’);

    %Guarda el nombre de la función
    guidata (info1_Callback);
```

Botón “Notas”:

Este botón nos permite ir a otra interfaz, en la cual nos da información sobre nuestro GUI, tal y como muestra la figura 35.

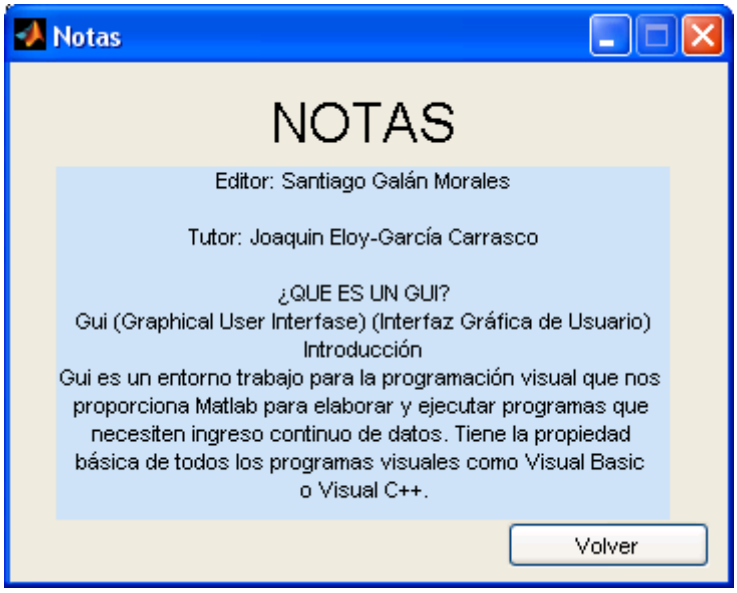

Figura 35. GUI Notas.

```
%Funciones determinadas por Matlab
Function varargout = Notas (varargin) gui_Singleton=1;
      gui_State = struct (‘gui_Name’, mfilename,
     ‘gui_Singleton’, gui_Singleton, ‘gui_OpeningFcn’,
     @Notas_OpeningFcn, ‘gui_OutputFcn’, @Notas_OutputFcn,
     ‘gui_LayoutFcn’, [], ‘gui_Callback’, []);

     if nargin && ischar (varargin{1})
          gui_State.gui_Callback = str2func (varargin {1});
     end

     if   nargout   [varargout{1:nargout}]   =   Gui_mainfcn
     (gui_State,varargin{:});
     else
     gui_mainfcn (gui_State, varargin{:});
     end

function Notas_OpeningFcn (hObject,handles,varargin)
     handles.output = hObject;
     guidata(hObject,handles);

%Esta función cierra el GUI Notas y abre el GUI PFCSGM
function Volver_Callback()
     close Notas
     PFCSGM
     guidata(Volver_Callback);
```

5.3 EXPLICACIÓN DEL PANEL DE CONTROL IZQ.

En la parte izquierda, vemos un panel controlado por sus botones o también llamados “pestañas” (marcado con la flecha en rojo), tal y como muestra la figura 36:

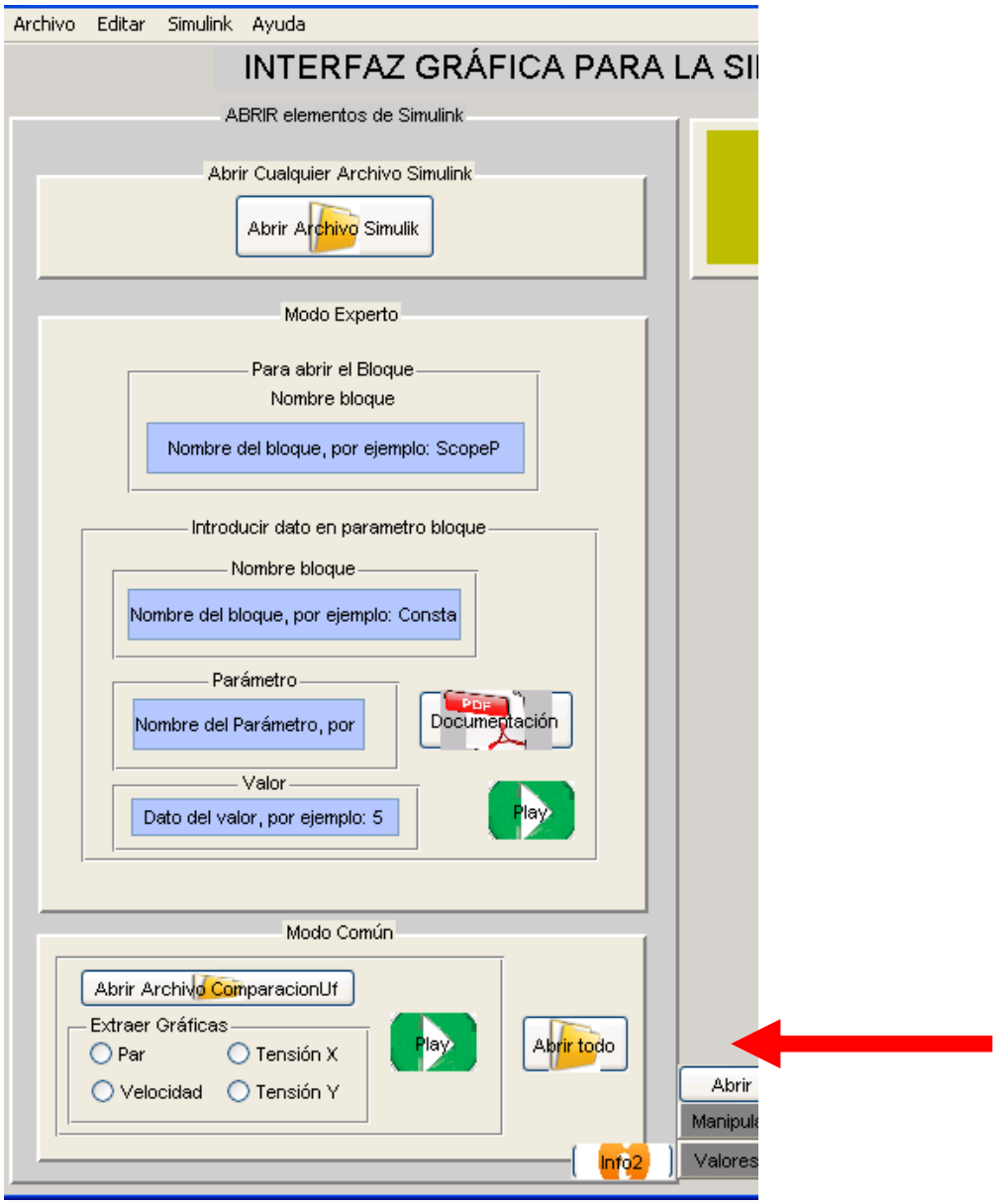

Figura 36. Panel de control Izq.

Esta parte es la más sería de nuestra interfaz, ya que aquí recae la mayor parte de trabajo. Tenemos tres paneles de control, de los cuales, dos están ocultos y uno está visible. Los botones o pestañas señalados por la flecha roja son los encargados de mostrar u ocultar los paneles de control. La pestaña seleccionada será la que muestre su panel de control respectivo. La pestaña “Abrir” se refiere al panel de control “Abrir archivo Simulink”. La pestaña “Manipular” se refiere al panel de control “manipulación

del archivo Simulink". La pestaña "Valores-I" se refiere al panel de control "Introducción de valores al archivo Simulink comparaciónUf".

En adelante explicaremos los tres paneles de control:

Antes de redactar los diferentes paneles de control existentes, vamos a exponer los botones o pestañas que controlan los paneles. Tenemos las pestañas del panel de control izq., tal y como muestra la figura 37:

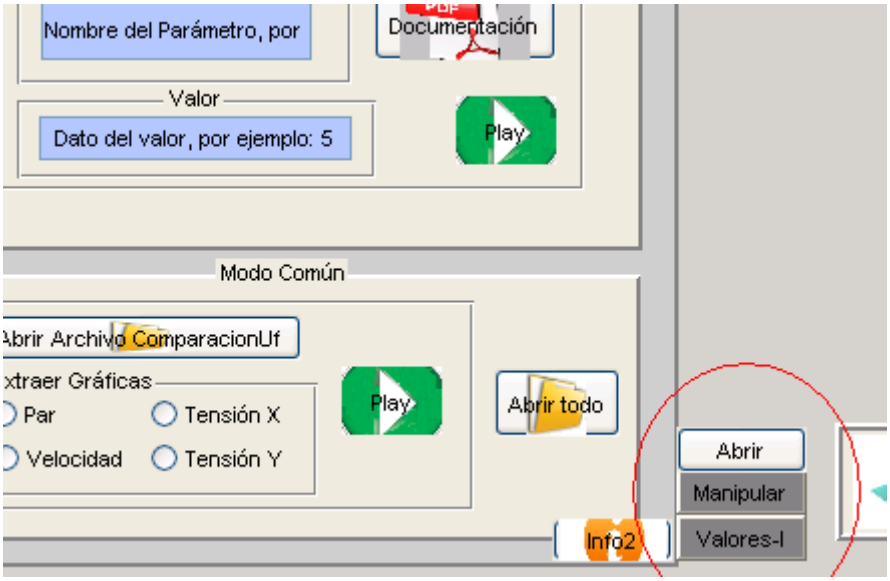

Figura 37. Pestañas del panel de control Izq.

Y tenemos las pestañas del panel de control dcha, tal como muestra la figura 38:

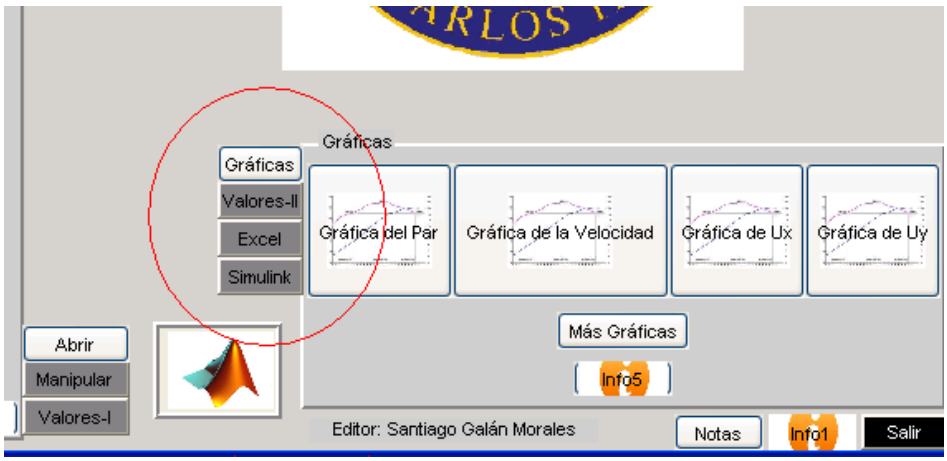

Figura 38. Pestañas del panel de control Dcha.

Como todos se programan de una manera similar, solo mencionaremos la pestaña “Abrir”, como muestra la figura 39.

Figura 39. Pestaña “Abrir”.

Esta pestaña tiene el fin de mostrar el panel de control “Abrir elementosSimulink” y ocultar el resto de paneles de control. Además para que facilite su visión, la pestaña seleccionada se cambiará el fondo de este botón a un color más iluminado pero las pestañas restantes tendrán un fondo de color oscuro para no resaltar.

La programación de la pestaña “Abrir” es:

```
%Función pestanaabrir: Esta función hace referencia al
botón pestaña "abrir".
function pestanaabrir_Callback (~, ~, handles)

     %Muestra un mensaje en el panel de texto general
          set (handles.text, ‘string’, ‘Está trabajando con
          el panel de control "ABRIR"’);

     %Hacemos visible el panel de control "valores", los
     demás los ocultamos
          Set (handles.panelmanipulacion, ‘Visible’,‘off’);
          Set (handles.panelintroval, ‘Visible’, ‘off’);
          Set (handles.panelvalores2, ‘Visible’, ‘on’);

     %Hacemos visible la pestaña "abrir", las demás las
     ocultamos
          set (handles.pestanamanipular, ‘backgroundcolor’,
          [0.5 0.5 0.5]);
          set (handles.pestanavalores1, ‘backgroundcolor’,
          [0.5 0.5 0.5]);
          set (handles.pestanaabrir, ‘backgroundcolor’,
          [0.925 0.914 0.847]);

     %Guardamos el nombre de la función
          guidata (pestanaabrir_Callback);
```

Panel de control: “Abrir archivo Simulink”, mostrado en la figura 40.

Figura 40. Panel de control izq.

Este panel es el encargado de abrir cualquier sistema Simulink, así como variar sus parámetros, en él nos encontramos tres sub-paneles:

1. El primer sub-panel que nos encontramos se llama “abrir archivo Simulink”, el cual se muestra en la figura 41:

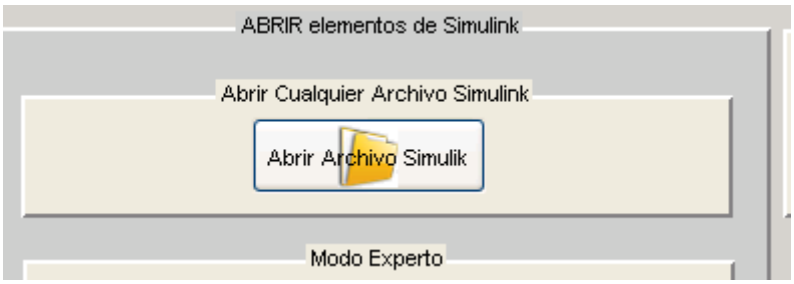

Figura 41. Botón Abrir

En este sub-panel, solo nos encontramos un botón, “abrir archivo Simulink”, el cual nos permite abrir cualquier archivo Simulink desde una ventana emergente, en la cual abre un directorio específico donde podemos encontrar los archivos Simulink y buscar el archivo requerido. Ventana emergente mostrada en la figura 42:

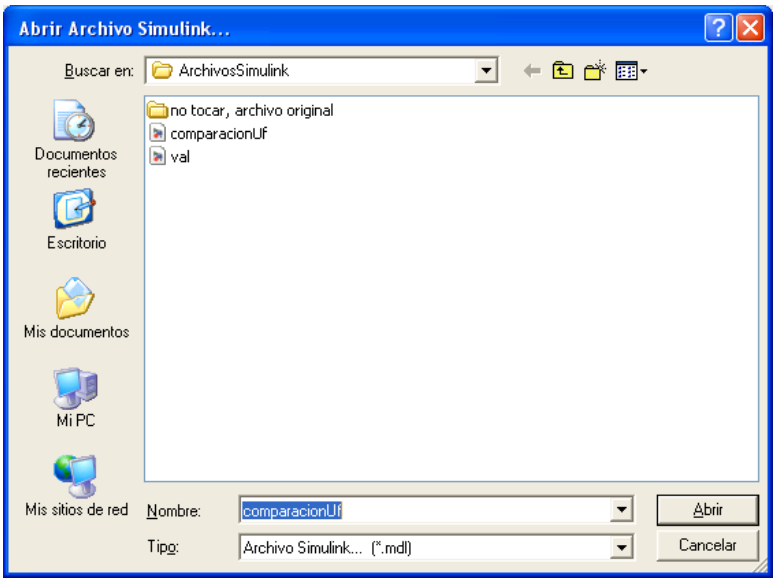

Figura 42. Ventana emergente, directorio.

Su programación es igual a la del menú editor superior de selección “abrir…” con la peculiaridad que la extensión requerida para abrir arhivos Simulink es (*.mdl).

2. El segundo sub-panel que nos encontramos se llama “Modo experto” y que se muestra en la figura 43:

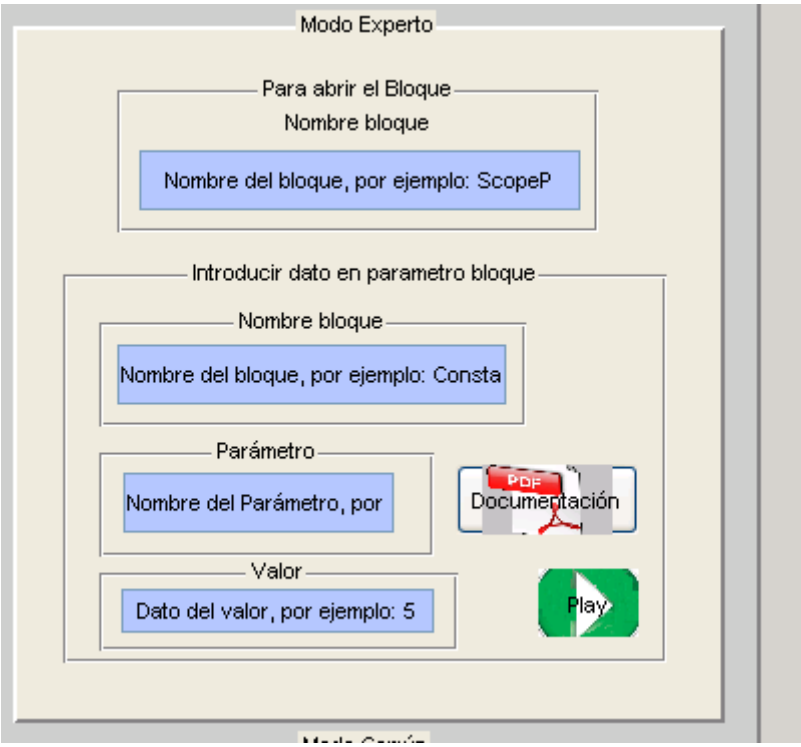

Figura 43. Modo Experto.

Este sub-panel se llama así por que el usuario tiene que tener experiencia en Simulink ya que nos encontramos con elementos un poco más específicos.

Una de las peculiaridades de Simulink es que está compuesto por bloques, pues bien, sabiendo esto, podemos comentar el primer elemento de este sub-panel, mostrado en la figura 44:

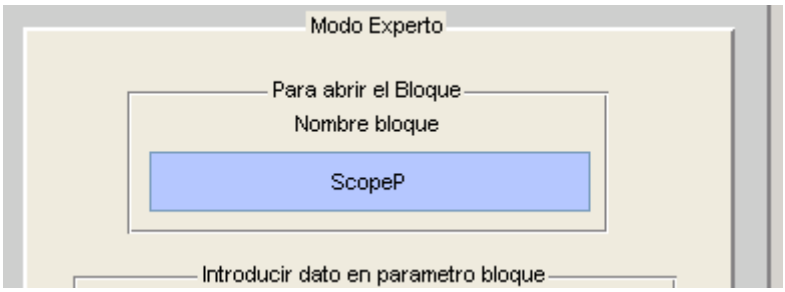

Figura 44. Modo Experto.

Este elemento nos permite abrir cualquier bloque que se encuentre en el archivo Simulink. Si por ejemplo hemos abierto el archivo Simulink "comaparcionUf", con anterioridad, tal y como se muestra la figura 45:

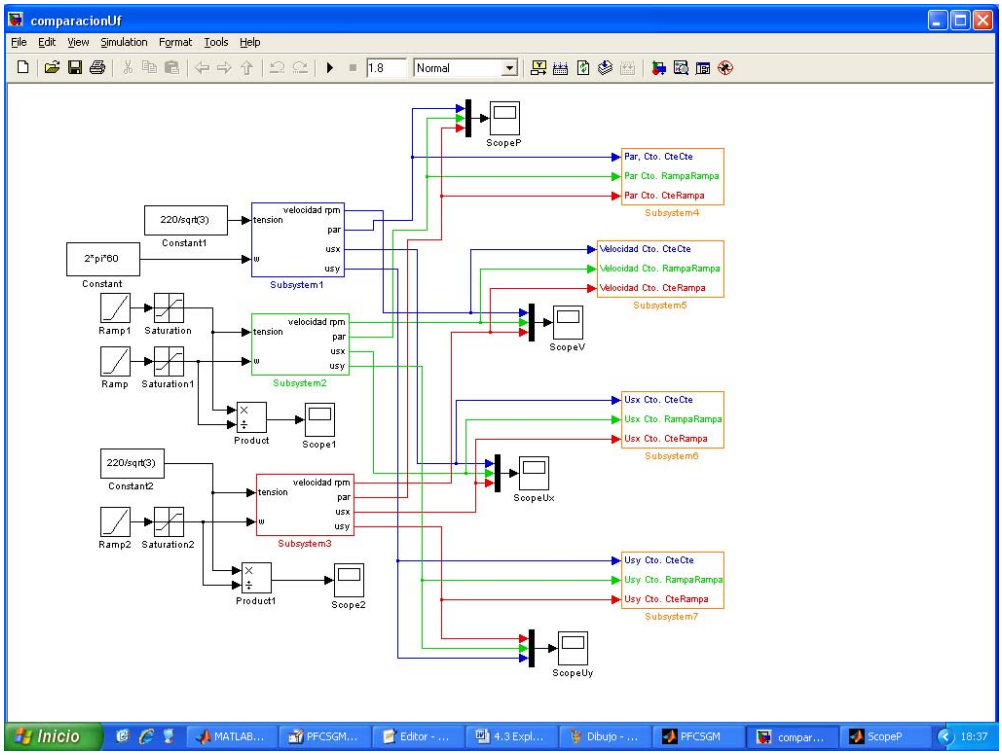

Figura 45. Archivo Simulink.

Y en el elemento de nuestra interfaz escribimos "ScopeP", abrirá el bloque mencionado, el cual es una ventana emergente que muestra la comparativa de las señales del par, tal y como se muestra la figura 46:

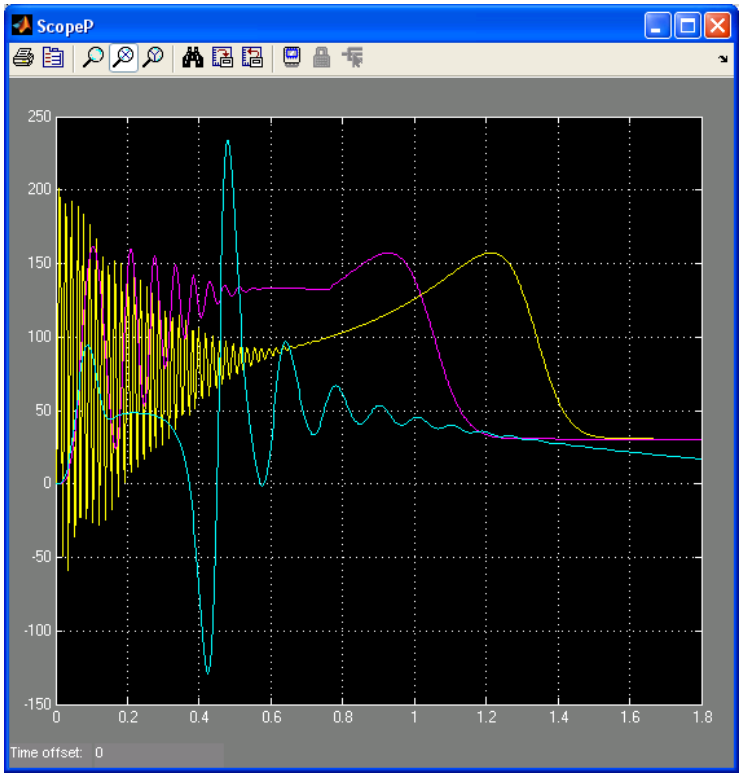

Figura 46. Scope

Para poner otro ejemplo diferente, podríamos escribir el nombre del bloque “Constant1”, en este caso se abrirá una ventana emergente donde muestra los parámetros, con sus respectivos valores, de este bloque mencionado, con la capacidad de poder variar estos valores, tal y como se muestra la figura 47:

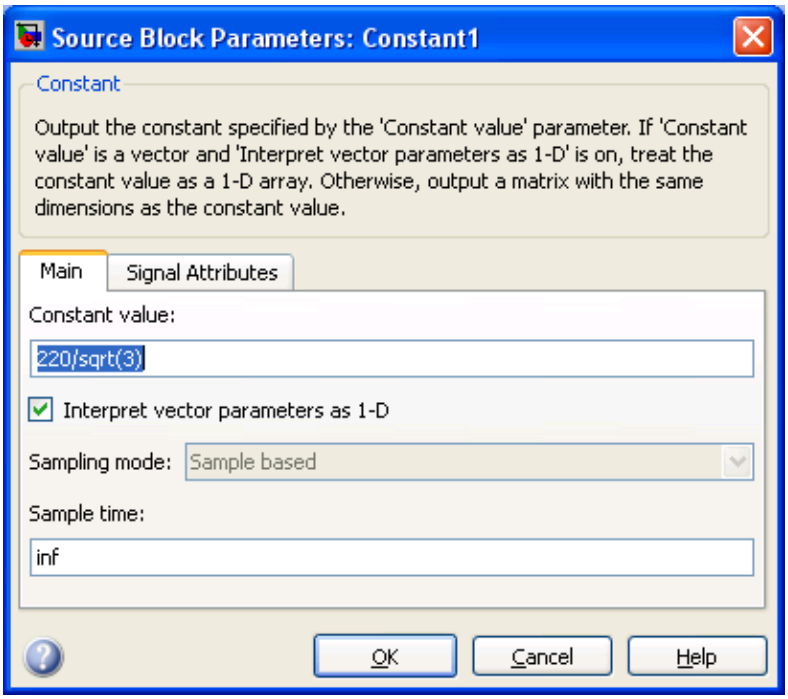

Figura 47. Bloque Constant1

La programación de este editor es:

```
%Función EditBloque: Esta función nos permite abrir cualquier bloque del archivo Simulink que hayamos abierto, con tan solo escribir el nombre del bloque
     function EditBloque_Callback (hObject, ~, handles)

     %Se cambia la figura del puntero del ratón por un reloj de espera
          Set (handles.figure1, 'Pointer', 'watch');

     %Guarda el valor introducido en la variable "nomsim"
          nomsim = get (hObject, 'string');

     %En el identificador "handles.ruta" tenemos guardado el nombre del archivo Simulink abierto, strcat nos junta el nombre del archivo con el nombre del bloque y lo manda a la variable
          ar = strcat (handles.ruta, '/', nomsim);

     %Abre el archivo Simulink guardado en la variable "ar"
          open_system (ar);

     %Muestra un mensaje en el panel de texto general, en este caso es el nombre del archivo Simulink
          Set (handles.text, 'string', ['Acaba de abrir el bloque:', ar]);

     %Cuando presiono el botón este se queda inactivo y la flecha del cursor se vuelve relojito de espera hasta el final que vuelve a ser como al principio
          Set (handles.figure1, 'Pointer', 'arrow');

     %Guardamos el nombre de la función
          Guidata (EditBloque_Callback);
```

Hasta aquí, ya hemos logrado variar los valores de los parámetros de cualquier bloque de cualquier archivo Simulink, pero de una manera en la que trabajas con varias ventanas. Los siguientes elementos, tienen la finalidad de hacerlo todo trabajando únicamente con nuestra interfaz.

Mostramos los siguientes elementos en la figura 48:

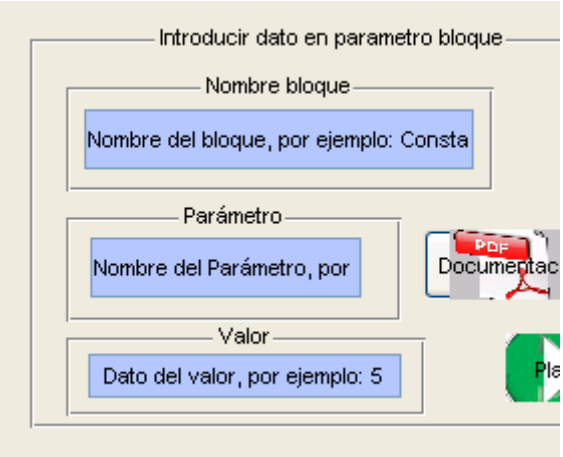

Figura 48. Introducción de datos

En estos tres editores, podemos introducir el nombre del bloque cuyos valores queremos variar, podemos introducir el nombre del parámetro cuyos valores queremos variar y en el tercer elemento, que podemos introducir los valores que queremos variar. Este método necesita de un control mayor sobre el Simulink por parte del usuario del GUI.

La forma de programar el elemento editor “Nombre bloque” es:

```
%Función EditBloque2: Esta función hace referencia al
editor "editbloque2" que nos permite nombrar el bloque.
     function EditBloque2_Callback (hObject, ~, handles)

     %Se cambia la figura del puntero del ratón por un
     reloj de espera
          Set (handles.figure1, ‘Pointer’, ‘watch’);

     %Guarda el valor introducido en la variable "nomb"
          nomb = get (hObject, ‘string’);
          handles.nomb = nomb;
          guidata (hObject,handles);

     %Muestra en el panel de texto general el valor
     introducido
          Set (handles.text, ‘string’, [‘Acaba de
          seleccionar el bloque:’, nomb]);

     %Cuando presiono el botón este se queda inactivo y la
     flecha del cursor se vuelve relojito de espera hasta
     el final que vuelve a ser como al principio
          Set (handles.figure1, ‘Pointer’, ‘arrow’);

     %Guardamos el nombre de la función
          guidata (EditBloque2_Callback);
```

La forma de programar el elemento editor “Parámetro” es:

```
%Función EditParámetro: Esta función hace referencia al
editor "editparamerto" que nos permite nombrar el
parámetro.
function EditParámetro_Callback (hObject, ~, handles)

     %Se cambia la figura del puntero del ratón por un
     reloj de espera
          Set (handles.figure1, 'Pointer', 'watch');

     %Guarda el valor introducido en la variable "para".
          para = get (hObject, 'String');
          handles.para = para;
          guidata (hObject, handles);

     %Muestra en el panel de texto general el valor
     introducido
          Set (handles.text,'string',['Acaba de seleccionar
          el parámetro:', para]);

     %Cuando presiono el botón este se queda inactivo y la
     flecha del cursor se vuelve relojito de espera hasta
     el final que vuelve a ser como al principio
          Set (handles.figure1, 'Pointer', 'arrow');

     %Guardamos el nombre de la función
          guidata (EditParámetro_Callback);
```

La forma de programar el elemento editor "valor" es:

```
%Función EditValor: Esta función hace referencia al editor
"editvalor" que nos permite nombrar el valor.
function EditValor_Callback (hObject, ~, handles)

     %Se cambia la figura del puntero del ratón por un
     reloj de espera
          set (handles.figure1, 'Pointer', 'watch');

     %Guarda el valor introducido en la variable "para"
          Val = get (hObject, 'String');
          handles.val = val;
          guidata (hObject, handles);

     %Muestra en el panel de texto general el valor
     introducido
          Set (handles.text,'string', ['Acaba de introducir
          el valor:', val]);

     %Cuando presiono el botón este se queda inactivo y la
     flecha del cursor se vuelve relojito de espera hasta
     el final que vuelve a ser como al principio
```

```
        Set (handles.figure1, 'Pointer', 'arrow');

    %Guardamos el nombre de la función
        guidata (EditValor_Callback);
```

Todo esto no valdría de nada sin el elemento "Play", mostrado en la figura 49:

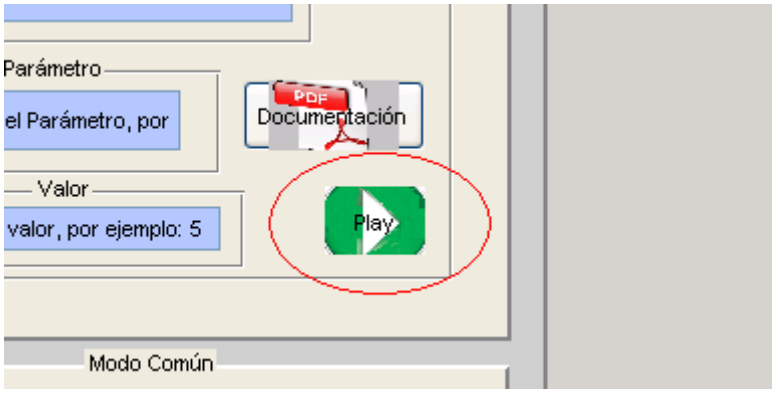

Figura 49. Botón "Play"

El botón "Play", nos permite actualizar los valores que hemos cambiado en el archivo Simulink. Si hemos abierto un bloque muy característico de Simulink, como es el bloque "SCOPE", el cual muestra las señales, el botón "Play" es indispensable para reproducir las señales respectivas.

Su programación es:

```
%Función Botonplay1: Esta función hace referencia al botón
"Play", el cual nos permite ejecutar el archivo Simulink
abierto, es decir, pulsar el run.
function Botonplay1_Callback (~, ~, handles)

    %Se cambia la figura del puntero del ratón por un
    reloj de espera
        Set (handles.figure1, 'Pointer', 'watch');

    %Inhabilita el botón del "Play"
        set(handles.BotonPlay1, 'Enable', 'off');

    %Muestra un mensaje en el panel de texto general
        Set (handles.text, 'string', 'Ha seleccionado
        "PLAY"');

   %Barra de espera
   %Abre o actualiza una ventana de diálogo de la barra
   de espera
        h_waitbar = waitbar (0, 'Por favor, espere...',
        'Position', [337, 430, 270, 50], 'Tag',
```

```
      'close_waitbar', 'CloseRequestFcn',
      @close_waitbar_CloseRequestFcn);

%Edita la barra de espera 300 veces
      for i = 1:300

%Determina si la entrada es válida
      if ishandle (h_waitbar)

            %Si es válida, repite la imagen e incrementa
            la unidad (i)
            waitbar (i / 300, h_waitbar)
            else

            %Si no es válida, retorna al GUI
            break
      end

 %Cierra la ventana
      delete (h_waitbar)

 %Habilita de nuevo el botón "Play"
      set (handles.Boton "Play", 'Enable', 'on');

 %Devuelve la figura original al puntero del ratón
      set (handles.figure1, 'Pointer', 'arrow');
```

```
%En handles.ruta tenemos guardado el nombre del
archivo Simulink abierto, strcat nos junta el nombre
del archivo con el nombre del bloque y lo manda a la
variable
     arc = strcat (handles.ruta, '/', handles.nomb);

%Como ejemplo a lo que sigue (no borrar): set_param
('val/sin', 'amplitude', '5');
     set_param (arc, handles.para, handles.val);

%Actualiza el archivo Simulink abierto
     set_param (gcs, 'SimulationCommand', 'Start')

%Salva el archivo Simulink modificado
     save_system (handles.FileName);

%Guardamos el nombre de la función
     guidata (Boton Play_Callback);
```

Si nos fijamos bien también nos podemos encontrar otro botón llamado "Documentación ", tal y como se muestra la figura 50.

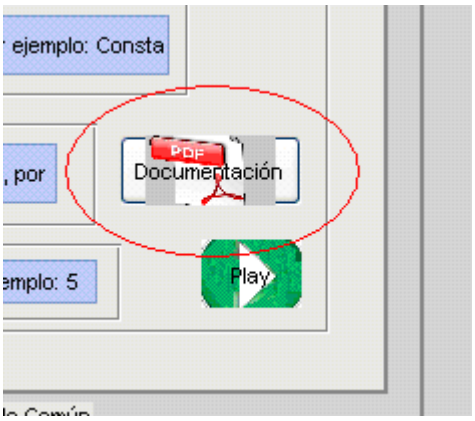

Figura 50. Botón "Documentación "

El botón "Documentación" nos permite abrir archivos de documentación, los cuales nos permiten documentarnos y recabar información sobre los principales bloques existentes en el archivo Simulink "ComparaciónUf ".

Cuando accionamos el botón "documentación", salta una ventana emergente, la cual abre un directorio específico donde se guardan esos documentos, tal y como se muestra la figura 51:

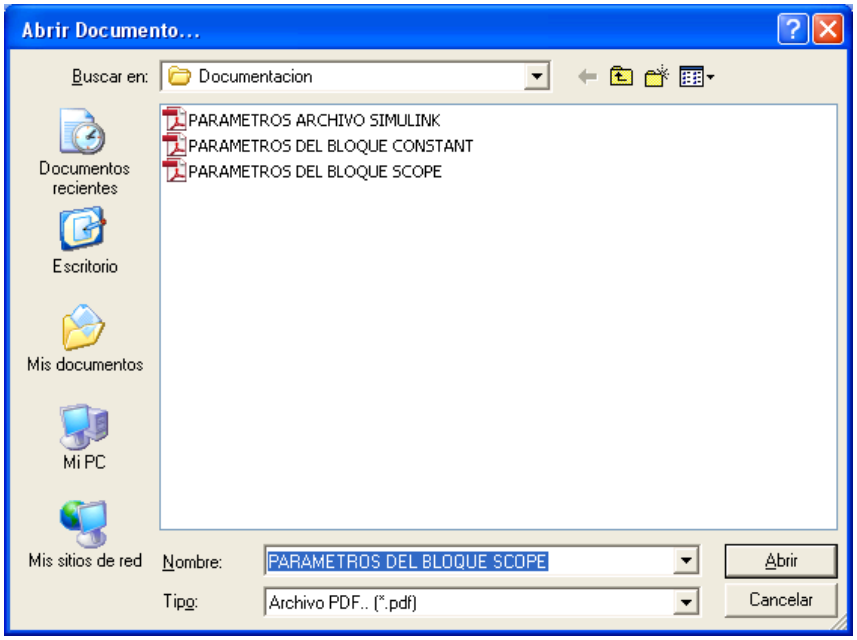

Figura 51. Elegir documentación

Su programación es muy similar al menú editor “abrir”.

Y por último, nos encontramos con el tercer sub-panel llamado “Modo común”, mostrado en la figura 52:

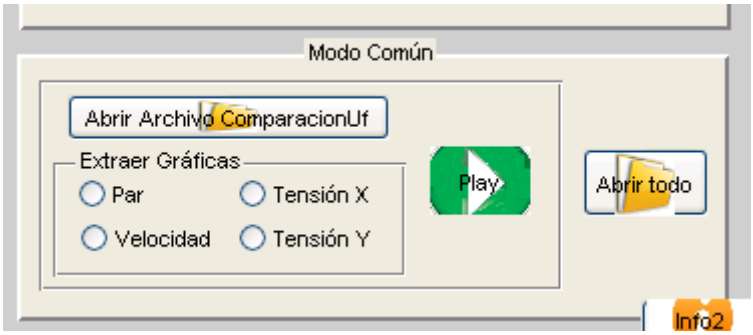

Figura 52. Modo común

Este sub-panel nos permite centrar nuestro trabajo en el archivo Simulink “ComparaciónUf “, de modo que nos sea más rápido, más cómodo y más eficaz nuestro trabajo.

En este sub-panel nos encontramos el botón “abrir archivo ComparaciónUf”, el cual nos abre directamente el archivo Simulink “ComparaciónUf” sin tener que buscarlo en algún directorio

Una vez conocidos estos archivos Simulink, vemos que tiene cuatro gráficas, pues bien, en nuestro GUI, dentro de este sub-panel, vemos que hay cuatro “RadioButton”, los cuales controlan, cada uno, un scope. Estos “RadioButton” están programados para que puedan abrirse independientemente de los demás. Cuando seleccionamos uno este abre una ventana emergente donde irá el scope.

Por último, nos encontramos dentro de este sub-panel, dos botones. El primero es el botón “Play” que como hemos mencionado antes, es capaz de actualizar el archivo Simulink abierto

Después tenemos el botón “abrir todo” que es capaz de abrir el archivo “comparaciónUf” y todos sus scopes, con el fin de hacer aún más rápido el trabajo del usuario.

El botón “Abrir Archivo ComparaciónUf” está mostrado en la figura 53:

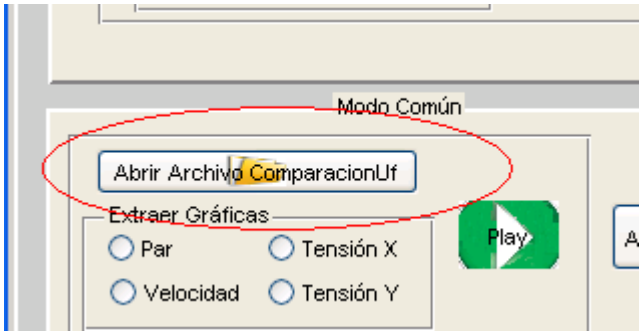

Figura 53. Y su programación es:

```
%Función BotonabrirArcComp: Esta función hace referencia al
botón “botonabrirarccomp” que nos permite trabajar mucho
más fácil con el archivo Simulink “comparaciónUf”. La
acción de este botón es abrir directamente este archivo.
     Function BotonabrirArcComp_Callback(hObject,~,handles)

     %Se cambia la figura del puntero del ratón por un
     reloj de espera
          Set (handles.figure1, ‘Pointer’, ‘watch’);

     %Inhabilita el botón del “Play”
          Set (handles.BotonabrirArcComp, ‘Enable’, ‘off’);

     %Barra de espera
     %Abre o actualiza una ventana de diálogo de la barra
     de espera
          h_waitbar=waitbar  (0,  ‘Por  favor,  espere...’,
          ‘Position’,[337,     430,     270,     50],‘Tag’,
          ‘close_waitbar’,                ‘CloseRequestFcn’,
          @close_waitbar_CloseRequestFcn);

     %Edita la barra de espera 300 veces
          for i = 1:300

     %Determina si la entrada es válida
          if ishandle (h_waitbar)
```

```
            %Si es válida, repite la imagen e incrementa
            la unidad (i)
            waitbar (i / 300, h_waitbar)
      else
            %Si no es válida, retorna al GUI
            break
            end
      end

%Verificar que existe el archivo Simulink
“ComparaciónUf”
      find_system(‘Name’,‘ArchivosSimulink\ComparaciónU
      f’);

%Abre el archivo Simulink “ComparaciónUf”
      open_system (‘ArchivosSimulink\ComparaciónUf’);

%Cierra la ventana
      delete (h_waitbar)

%Variable para que vuelva a abrirse el archivo
“ComparaciónUf”
      valcond = 1;
      handles.valcond = valcond;
      guidata (hObject,handles);

%Muestra un mensaje en el panel de texto general
      set (handles.text, ‘string’, ‘Acaba de abrir el
      archivo Simulink ComparaciónUf.mdl’);

%guarda nombre del archivo para después cerrarlo
      handles.FileName = ‘ComparaciónUf.mdl’;
      guidata (hObject,handles);

%Escribir en la variable handles.ruta el nombre de
“comparaciónUf” para que el botón salir guarde el
archivo
      handles.ruta = ‘ComparaciónUf’;
      guidata (hObject,handles);

%habilita de nuevo el botón “Play”
      set (handles.BotonabrirArcComp, ‘Enable’, ‘on’);

%devuelve la figura original al puntero del ratón
      set (handles.figure1, ‘Pointer’, ‘arrow’);

%guardamos el nombre de la funcion
      guidata (BotonabrirArcComp_Callback);
```

Los “RadioButton” se muestran en la figura 54:

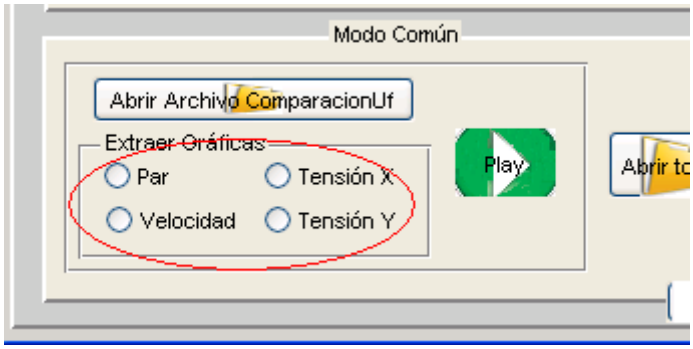

Figura 54. Figura de los RadioButton ‘s

La programación de uno de ellos es:

```
%Función radiobuttonpar: Esta función hace referencia a un
radiobutton, el cual abre la gráfica comparativa del par
     function radiobuttonpar_Callback (hObject,~,handles)

     %Se cambia la figura del puntero del ratón por un
     reloj de espera
          set (handles.figure1, ‘Pointer’, ‘watch’);

     %Condición de acción
          if (get (hObject, ‘Value’) == get(hObject,
          ‘Max’))

          %Si accionamos el "radiobutton", seleccionamos la
          opción de abrir el gráfico
          %Muestra un mensaje en el panel de texto general
          Set (handles.text, ‘String’, ‘Ha seleccionado la
          opción de abrir el Gráfico del Par’)

          %Abre la ventana grafica del par
          set_param (‘ComparaciónUf/ScopeP’, ‘open’, ‘on’);

          else

     %Si deseleccionamos el "radiobutton", seleccionamos la
     opción de cerrar el gráfico, muestra un mensaje en el
     panel de texto general
          Set (handles.text, ‘String’, ‘Ha seleccionado la
          opción de cerrar el gráfico del par’)

     %Abre la ventana grafica del par
          set_param (‘ComparaciónUf/ScopeP’, ‘open’,
          ‘off’);
          end
```

```
%Devuelve la figura original al puntero del ratón
     Set (handles.figure1, 'Pointer', 'arrow');

%Guardamos el nombre de la funcion
     Guidata (radiobuttonpar_Callback);
```

La programación del botón "Play" es similar al anterior botón "Play" y el botón "Abrir Todo el Archivo comparaciónUf" es el conjunto del botón "Abrir archivo ComparaciónUf" y abrir los cuatro radiobutton's.

Y por último tenemos el botón "info2", el cual nos muestra un mensaje de texto en el panel de control general, dándonos información sobre este panel de control "abrir archivo de Simulink", tal y como se muestra en la figura 55:

Figura 55. Info

Para ver el siguiente panel de control (Panel de control "Manipular") debemos pulsar el botón o pestaña "Manipular", tal y como se muestra en la figura 56:

Figura 56. Botón o pestaña manipular

Su programación es similar a la de todas las pestañas:

```
%Función "pestanamanipular", hace referencia al botón
pestaña "manipular".
     function pestanamanipular_Callback (~, ~, handles)

     %Muestra un mensaje en el panel de texto general  .
          Set (handles.text, 'string', 'Está trabajando con
          el panel de control "MANIPULAR"');

     %Hacemos visible el panel de control "Manipulación",
     los demás los oculta.
          Set (handles.panelmanipulación, 'Visible', 'on');
          Set (handles.panelintroval, 'Visible', 'off');
          Set (handles.panelvalores2, 'Visible', 'off');
```

```
%Hacemos visible la pestaña "manipulación", las demás
las ocultamos
     set (handles.pestanamanipular, 'backgroundcolor',
     [0.925 0.914 0.847]);
     set(handles.pestanavalores1,'backgroundcolor',
     [0.5 0.5 0.5]);
     set (handles.pestanaabrir, 'backgroundcolor',
     [0.5 0.5 0.5]);

%Guardamos el nombre de la función
     guidata (pestanamanipular_Callback);
```

Con el cual aparecerá el panel de control manipulación de la ventana de trabajo Simulink, tal y como se muestra en la figura 57:

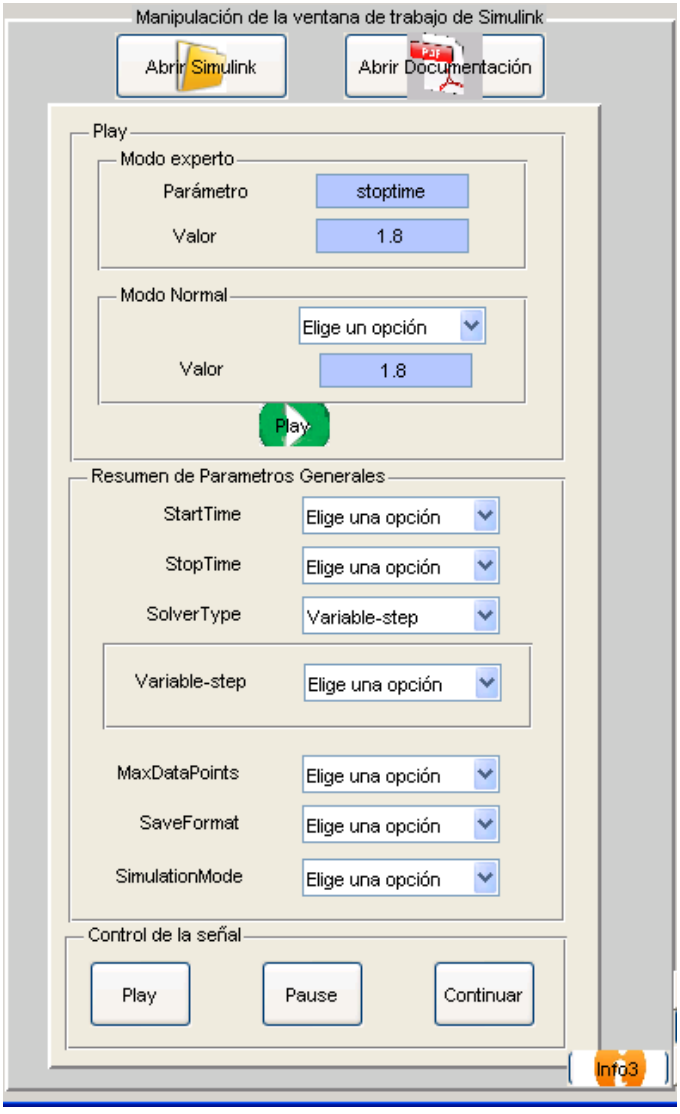

Figura 57. Panel de control, Manipular

Este panel de control nos permite variar los parámetros de la ventana de trabajo Simulink. Así como variar los parámetros principales con más comodidad, rapidez y eficacia.

Ilustramos el elemento en la figura 58:

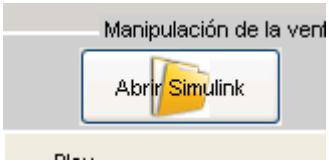

Figura 58. Botón abrir.

Cuya programación es similar a los botones de abrir pero con la extensión (*.mdl)

Lo primero que nos encontramos es el botón “Abrir”, que nos permite abrir cualquier archivo Simulink, si es que en el anterior panel de control no habíamos abierto el deseado o lo habíamos pasado por alto.

A la derecha de este botón “abrir”, tenemos el botón “documentación”, tal y como se muestra en la figura 59:

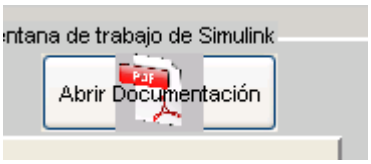

Figura 59. Botón documentación.

Cuya programación es similar a los botones de abrir pero con la extensión *.pdf

Este botón nos permite abrir documentación para recabar información sobre la ventana de trabajo.

Inmediatamente después nos encontramos el sub-panel “Modo Experto”, tal y como se muestra en la figura 60:

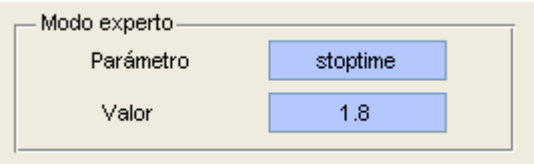

Figura 60. Modo experto.

Este sub-panel, nos da la capacidad de poner cualquier valor a cualquier parámetro de la ventana de trabajo de Simulink de una manera muy rápida y eficaz pero también requiere un conocimiento de la ventana de trabajo muy alto.

La programación del editor del parámetro es:

```
%Función editparámetro1: Esta función que hace referencia
al editor (editparámetro1) nombra al parámetro que queremos
variar
     function editparámetro1_Callback (hObject,~,handles)

     %Guardamos el dato en la variable
          para = get (hObject, 'string');
          handles.para = para;
          guidata (hObject, handles);

     %Muestra un mensaje en el panel de texto general
          Set (handles.text, 'string', ['Nombre del
          parámetro Simulink:', handles.para])

     %Guardamos nombre función
          Guidata (editparámetro1_Callback);
```

La programación del editor del valor es:

```
%Función editvalor1: Esta función que hace referencia al
editor (editvalor1) nombra el nuevo valor.
     function editvalor1_Callback (hObject, ~, handles)

     %Guardamos el dato en la variable
          sto = get (hObject, 'string');
          handles.sto = sto;
          guidata (hObject, handles);

     %Muestra un mensaje en el panel de texto general
          set (handles.text, 'string', ['Ha introducido el
          valor:', handles.sto])

     %Guardamos nombre función
          guidata (editvalor1_Callback);
```

Una vez introducidos los valores, debemos pulsar el botón "Play", para ingresarlos en el archivo Simulink, cuya programación es similar a los ya mencionados botones "Play".

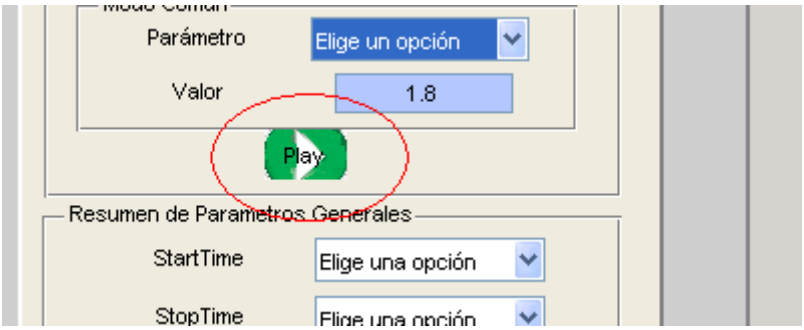

Figura 61. Botón “Play “

Después nos encontramos el sub-panel “Modo Común”, tal y como se muestra en la figura 62:

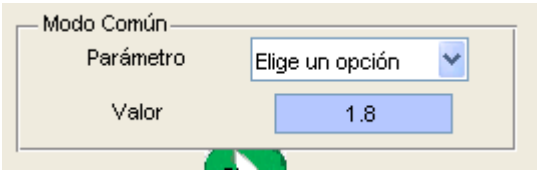

Figura 62. Modo común.

Este sub-panel, nos da la capacidad de poner cualquier valor a los principales parámetro de la ventana de trabajo de Simulink de una manera más rápida y eficaz y que requiere menos conocimientos de la ventana de trabajo.

Imagen del despliegue de opciones de selección se muestra en la figura63:

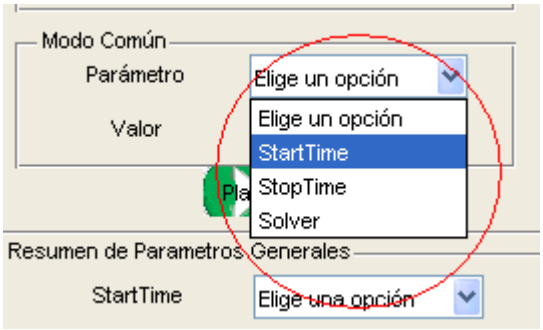

Figura 63. Despliegue de opciones.

Como vemos, podemos elegir entre los tres parámetros más utilizados. El editor de valor, podemos editar cualquier dato, para ello tenemos que tener un nivel básico del parámetro. La información que nos dan los documentos de apoyo será de gran ayuda.

La programación del “Pop-Menú Parámetro” es:

```
%Función PopupmenuGeneral: esta función hace referencia a
un elemento llamado "popmenu", en él podemos elegir entre
varias opciones de parámetros
     function PopupmenuGeneral_Callback (hObject,~,handles)

     %Guardamos la opción seleccionada en la variable
          fun = get (hObject, ‘Value’);

          %El comando switch determina que función será
          graficada.
               switch fun

          case 1

          %Muestra un mensaje en el panel de texto general
          Set (handles.text, ‘string’, ‘Elige un opción’);

          case 2

          %Muestra un mensaje en el panel de texto general
               Set (handles.text, ‘string’, ‘Ha elegido el
               parámetro StartTime’);

          %Guardamos el dato en la variable
               handles.para = ‘StartTime’;
               guidata (hObject,handles);

          case 3

          %Muestra un mensaje en el panel de texto general
               Set (handles.text, ‘string’, ‘Ha elegido el
               parámetro StopTime’);

          %Guardamos el dato en la variable
               handles.para=‘StopTime’;
                guidata(hObject,handles);

          case 4

          %Muestra un mensaje en el panel de texto general
               Set (handles.text, ‘string’, ‘Ha elegido el
               parámetro Solver’);

          %Guardamos el dato en la variable
               handles.para = ‘Solver’;
               guidata (hObject,handles);
          end

          %Guardamos el nombre de la función
               guidata (PopupmenuGeneral_Callback);
```

La programación del editor “Valores” es:

```
%Función editvalormanipular: Esta función que hace
referencia al editor (editvalormanipular) nombra el nuevo
valor.
     Function editvalormanipular_Callback (hObject, ~,
     handles)

     %Guardamos el valor introducido en una variable
          sto = get (hObject, ‘string’);
          handles.sto = sto;
          guidata (hObject, handles);

     %Muestra un mensaje en el panel de texto general
          set (handles.text, ‘string’, [‘Ha seleccionado el
          valor:’, handles.sto])

     %Guardamos nombre función
          guidata (editvalormanipular_Callback);
```

Una vez introducidos los valores, debemos pulsar el botón “Play”, para ingresarlos en el archivo Simulink, cuya programación es similar a los ya mencionados botones “Play”, cuya imagen muestra la figura 64.

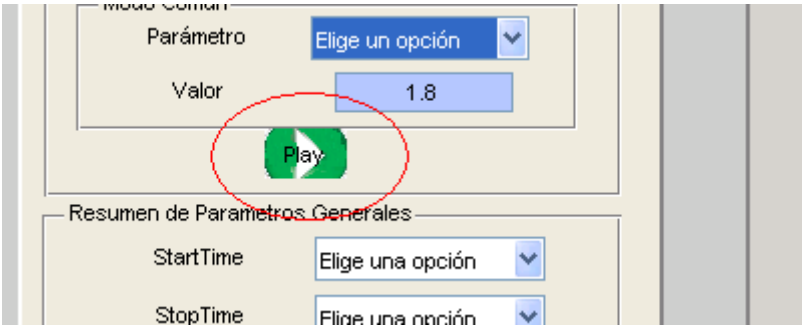

Figura 64. Botón “Play”

El siguiente sub-panel que nos encontramos es un resumen de los principales parámetros con varias opciones para los valores, la imagen es mostrada en la figura 65:

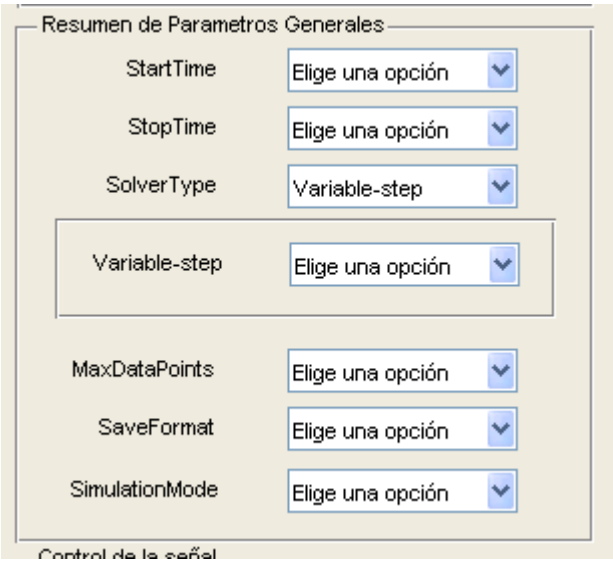

Figura 65. Resumen de parámetros.

La forma de programar es muy similar al " Pop-Menú Parámetro ", la unica distinción que existe en este apartado está en la elección del parámetro "Solver-Step " que según la variable que elijamos nos muestra un sub-panel de control "Variable-step" o el otro sub-panel de control "Fixed-step".

La programación del parámetro "Solver Type" es:

```
%Función PopupmenuSolverType: Esta función hace referencia
al popmenu del parámetro "SolverType"
     function  PopupmenuSolverType_Callback  (hObject,  ~,
     handles)

     %Guardamos el dato en la variable
          handles.para = 'SolverType';
          guidata (hObject, handles);

     %Guardamos la opción seleccionada en la variable
          fun4 = get (hObject, 'Value');

               %El  comando  switch  determina  que  función
               será graficada.
               switch fun4

          case 1

               %Guardamos la variable
               handles.sto = 'Variable-step';

               %Muestra  un  mensaje  en  el  panel  de  texto
               general  Set  (handles.text,  'string',  'Ha
               Elegido la variable Variable-step');
```

```
        %Escondemos el panel de control de la
        variable variable-step y mostramos el panel
        de control de la variable Fixed-step
        Set (handles.uipanel55, 'Visible', 'on');
        Set (handles.uipanel56, 'Visible', 'off');
    case 2

        %Guardamos la variable
        handles.sto = 'Fixed-step';

        %Muestra un mensaje en el panel de texto
        general
        Set (handles.text, 'string', 'Ha Elegido la
        variable Fixed-step');

        %Escondemos el panel de control de la
        variable Variable-step y mostramos el panel
        de control de la variable Fixed-step
        set(handles.uipanel55,'Visible','off');
        set(handles.uipanel56,'Visible','on');
    end

        %Guardamos el nombre de la función
        guidata (PopupmenuSolverType_Callback);
```

Y por último nos encontramos el sub-panel de "control de señal", tal y como se muestra en la figura 66:

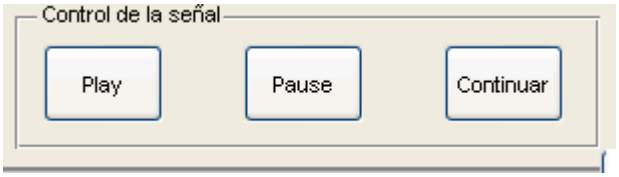

Figura 66. Control de señal.

Este sub-panel, es capaz de reproducir señales y pausarlas para su estudio. Estas señales se reproducen en los bloques de Simulink llamados "Scope".

Programación del botón "Play ".

```
%Función PlaySimulink: Esta función que hace referencia al
botón play, el cual nos permite ejecutar el archivo
Simulink abierto, es decir, pulsar el run.
    function PlaySimulink_Callback (~,~,handles)

    %Muestra un mensaje en el panel de texto general
        set(handles.text, 'string', 'Ha pulsado "Play",
        se actualizarán los valores y se representarán
        las gráficas');
```

```
        %Representamos la gráfica
             set_param (handles.ruta,
             'SimulationCommand', 'start');

        %Guardamos nombre función
             Guidata (PlaySimulink_Callback);
```

Programación del botón "Pause ".

```
%Función PauseSimulink: Esta función que hace referencia al
botón pause, el cual nos permite gráficas.
     function PauseSimulink_Callback (~, ~, handles)

     %Muestra un mensaje en el panel de texto general
          set (handles.text, 'string', 'Ha pulsado "Pause",
          las señales se pausarán para su estudio');

     %Pausamos la representación de la gráfica
          set_param(handles.ruta,      'SimulationCommand',
          'pause');

     %Guardamos nombre función
          guidata (PauseSimulink_Callback);
```

Programación del botón "Continuar ".

```
%Función   ContinuarSimulink:   Esta   función   que   hace
referencia al botón continue, el cual nos permite gráficas.
     function ContinuarSimulink_Callback (~,~,handles)

     %Muestra un mensaje en el panel de texto general
          set(handles.text,'string','Ha             pulsado
          "Continuar",      las      señales      seguirán
          reproduciéndose');

     %Continuamos con la representación de la gráfica
          Setparam   (handles.ruta,   'SimulationCommand',
          'continue');

     %guardamos nombre función
          guidata (ContinuarSimulink_Callback);
```

Y por último nos encontramos el botón info, mostrado en la figura 67.

Figura 67. Botón info.

Si por último, pulsamos en la pestaña o botón “Valores - I”, visualizamos el panel de control “Valores”. Este panel de control es exclusivo para el control del archivo Simulink “ComparaciónUf “. Este panel nos permite introducir valores a todas las entradas de este archivo Simulink, tal como se muestra en la figura 68.

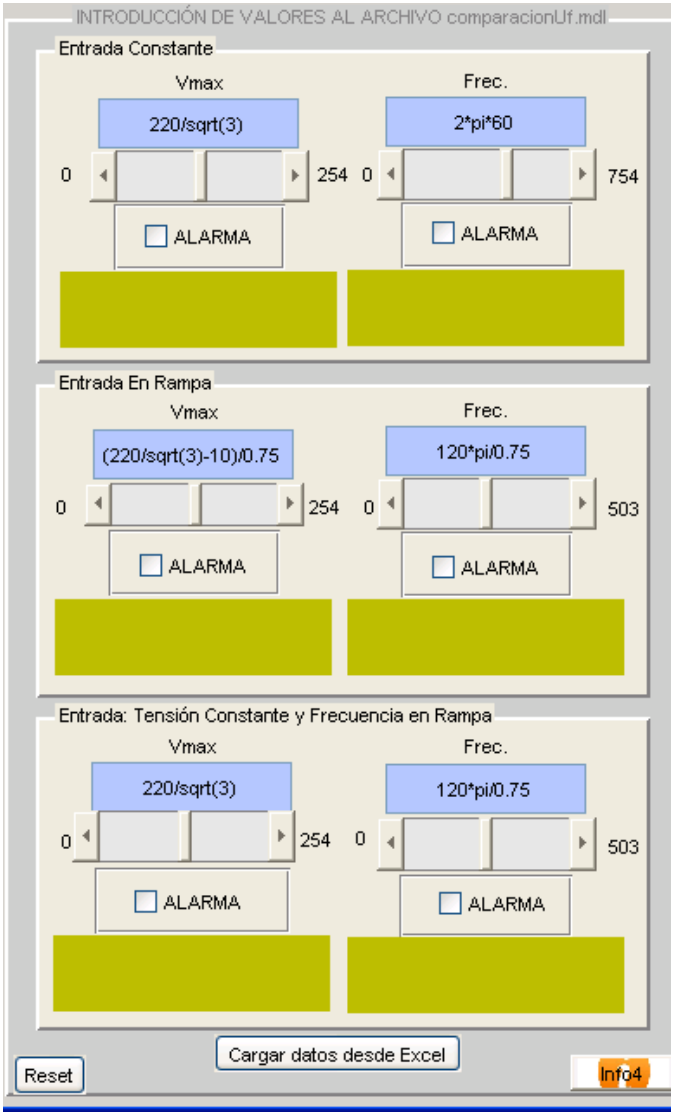

Figura 68. Introducción de Valores

Las entradas son los parámetros Vmax. y Frecuencia en cada uno de los tres circuitos (Circuito Constante-Constante, Circuito Constante-Rampa y Circuito Rampa-Rampa).

Nos fijaremos en el primer sub-panel, Circuito Constante-Constante, ya que los siguientes circuitos son similares, tal como se muestra en la figura 69:

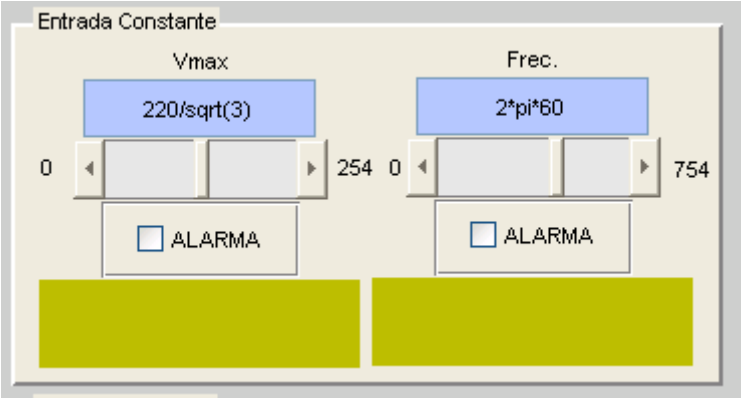

Figura 69. Circuito Constante-Constante.

Para introducir los datos podemos utilizar tanto el editor de texto como el "Slider" (deslizador) que constantemente visualizará su valor en el editor respectivo.

Cada parámetro tiene un rango de buen funcionamiento, fuera de él, el circuito está en peligro. Pues bien, una vez introducido un valor fuera del rango, se activará la alarma visual, tal como se muestra en la figura 70:

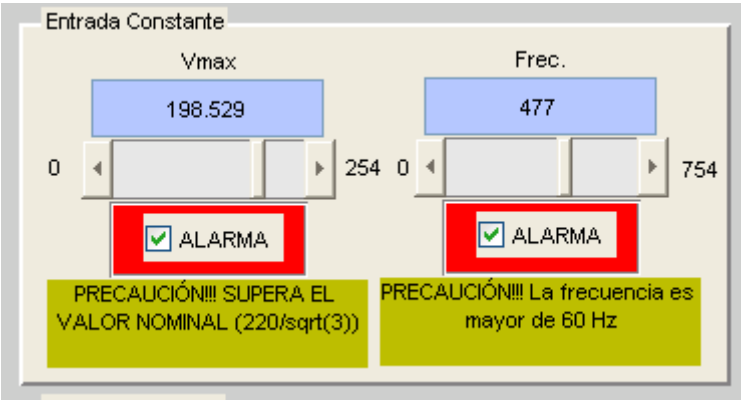

Figura 70. Alarma.

Como vemos, los CheckBox se activan, sus frames cambian de color y en el panel de texto local, te avisa del error.

La programación del editor Vmax. es:

```
%Función editC1: Esta función hace referencia a la entrada
Vmax circuito cte-cte
     function editC1_Callback (hObject, ~, handles)

     %Guarda el dato introducido en el editor en la
     variable "var1".
          handles.var1 = get (hObject, 'string');
          guidata (hObject, handles);

     %Introduce la variable "var1" en el sliderC1
          set(handle.sliderC1,'value',str2double(handles.va
          r1));

     %Muestra un mensaje en el panel de texto general
          Set (handles.text, 'string', ['Tensión=',
          handles.var1, 'Voltios'])

     %Guarda el dato en la variable del archivo Excel
          handles.arcexcelC1 = handles.var1;
          guidata (hObject, handles);

     %Condición de alarma
          if (str2double(handles.var1) < 128);

     %Si cumple con la condición, realiza: Muestra un
     mensaje en el panel de textoa, en este caso borra el
     texto de alarma y escribe un espacio en blanco
          Set (handles.texta, 'String', ' ');

     %Deshabilita el Check box
          Set (handles.alarma1, 'Value', 0);

     %Cambia el color del recuadro, tipo no llamativo
          set      (handles.frame1,      'Backgroundcolor',
          'factory')
          else

     %Si no cumple con la condición, realiza:
     %Muestra un mensaje de alarma en el panel de textoa
          set   (handles.texta,   'String',   'PRECAUCIÓN!!!
          SUPERA EL VALOR NOMINAL (220/sqrt(3))');

     %Habilita el Check box alarma1
          set (handles.alarma1, 'Value', 1);

     %El frame cambia ha estado de alarma
          Set (handles.frame1, 'Backgroundcolor', 'red')
          end
```

```
%Guardamos nombre función
     guidata (hObject,handles);

%Para pasar el dato al archivo Simulink:
     set_param (‘comparaciónUf/Constant1’, ‘value’,
     handles.var1);

%Start
     set_param (gcs, ‘SimulationCommand’, ‘Start’)

%Guardamos nombre función
     guidata (editC1_Callback);
```

La programación del “Slider” Vmax. es:

```
%Función editC2: Esta función hace referencia a la entrada
Frec circuito cte cte
     function editC2_Callback (hObject, ~, handles)

     %Guardo dato introducido en el editor en la variable
     "var1"
          handles.var1 = get (hObject, ‘string’);
          guidata (hObject, handles);

     %Introduzco "var1" en el sliderC1
          Set (handles.sliderC2, ‘value’, str2double
          (handles.var1));

     %Muestra un mensaje en el panel de texto general
          Set (handles.text, ‘string’, [‘frecuencia=’,
          handles.var1, ‘Rad/seg(2*pi*frec.)’])

     %Dato en edit text a stactic text
     %Guarda el dato en la variable del archivo Excel
          handles.arcexcelC2 = handles.var1;
          guidata (hObject, handles);

     %Condición de alarma
          if (str2double (handles.var1) < 314);

     %Si cumple con la condición, realiza:
     %Muestra un mensaje de alarma en el panel de textob
          Set   (handles.textb,   ‘String’,   ‘PRECAUCIÓN!!!
          SUPERA EL VALOR NOMINAL (220/sqrt(3))’);

     %Habilita el Check box alarma2
          set (handles.alarma2, ‘Value’, 1);

     %El frame cambia ha estado de alarma
          set (handles.frame2, ‘Backgroundcolor’, ‘red’)
```

```
     elseif (str2double (handles.var1) > 377);


%Si cumple con la condición, realiza:
%Muestra un mensaje de alarma en el panel de textob
     set   (handles.textb,   'String',   'PRECAUCIÓN!!!
     SUPERA EL VALOR NOMINAL (220/sqrt(3))');

%Habilita el Check box alarma2
     set (handles.alarma2, 'Value', 1);

%El frame cambia ha estado de alarma
     set (handles.frame2, 'Backgroundcolor', 'red')
     else

%Si no cumple con la condición, realiza:
%Muestra un mensaje en el panel de textob, en este
caso borra el texto de alarma y escribe un espacio en
blanco
     set (handles.textb, 'String', ' ');

%Deshabilita el Check box "alarma2".
     set (handles.alarma2, 'Value', 0);

%Cambia el color del recuadro, tipo no llamativo
     set      (handles.frame2,      'Backgroundcolor',
     'factory')
     end
     guidata (hObject, handles);

%Pasa el dato al archivo Simulink:
     set_param ('comparaciónUf/Constant', 'value',
     handles.var1);

%Start
     set_param (gcs, 'SimulationCommand', 'Start')

%Guardamos nombre función
     guidata (editC2_Callback);
```

La programación de los "Check-Box" consta de tan solo definir la función. Esto es por que tan solo son elementos de visualización y no de acción.

Para los paneles de texto ni tan siquiera hace falta nombrar sus funciones.

En la parte baja de este panel vemos dos botones muy importantes, tal como se muestra en la figura 71:

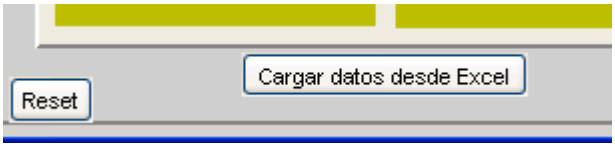

Figura 71. Botones “Reset” y “Cargar datos”.

Son los botones “Reset” y “Cargar datos”. El botón “Reset” devuelve a nuestro GUI y al archivo Simulink, los valores iniciales y el botón “Cargar datos” nos carga los datos que tenga el archivo Excel seleccionado en nuestro GUI desde una ventana emergente donde nos muestra un directorio y por tanto en el archivo Simulink. Existe un archivo creado por defecto que viene estructurado de una manera más factible y visual hacia el usuario (General Guardar y Cargar).

La programación del botón “Reset” es:

```
%Botón Reset: Nos devuelve los valores iniciales a nuestro
sistema Simulink “ComparaciónUf”, a los editores y a los
deslizadores de nuestro GUI pero además borra las alarmas y
por último restablece los valores iniciales a nuestro
archivo Simulink "ComparaciónUf"
     function reset_Callback (~, ~, handles)

     %Muestra un mensaje en el panel de texto general
     Set (handles.text, ‘string’, ‘Botón Reset: Restablece
     los valores predeterminados’);

     %Restablecemos los valores iniciales de entrada del
     archivo comparaciónUF
     Set (handles.editC1, ‘string’, ‘220/sqrt(3)’);
     Set (handles.editC2, ‘string’, ‘2*pi*60’);
     Set (handles.editR1, ‘string’, ‘(220/sqrt(3)-
     10)/0.75’);
     Set (handles.editR2, ‘string’, ‘120*pi/0.75’);
     Set (handles.editCR1, ‘string’, ‘220/sqrt(3)’);
     Set (handles.editCR2, ‘string’, ‘120*pi/0.75’);

     %Restablecemos los valores iniciales de los sliders
     del archivo comparaciónUF
     set (handles.sliderC1, ‘ Value’, 127);
     set (handles.sliderC2, ‘ Value’, 477);
     set (handles.sliderR1, ‘ Value’, 127);
     set (handles.sliderR2, ‘ Value’, 250);
     set (handles.sliderCR1, ‘ Value’, 127);
     set (handles.sliderCR2, ‘ Value’, 250);

     %Para borrar lo escrito en el static text y que no
     ponga que hay alarma
     set (handles.texta, ‘String’, ‘ ’);
     set (handles.textb, ‘String’, ‘ ’);
     set (handles.textc, ‘String’, ‘ ’);
```

```
set (handles.textd, ‘String’, ‘ ’);
set (handles.texte, ‘String’, ‘ ’);
set (handles.textf, ‘String’, ‘ ’);

%Desconecta la alarma
set (handles.alarma1, ‘ Value’, 0);
set (handles.alarma2, ‘ Value’, 0);
set (handles.alarma3, ‘ Value’, 0);
set (handles.alarma4, ‘ Value’, 0);
set (handles.alarma5, ‘ Value’, 0);
set (handles.alarma6, ‘ Value’, 0);

%Devuelve al estado original a los cuadros
set (handles.frame1, ‘Backgroundcolor’, ‘factory’)
set (handles.frame2, ‘Backgroundcolor’, ‘factory’)
set (handles.frame3, ‘Backgroundcolor’, ‘factory’)
set (handles.frame4, ‘Backgroundcolor’, ‘factory’)
set (handles.frame5, ‘Backgroundcolor’, ‘factory’)
set (handles.frame6, ‘Backgroundcolor’, ‘factory’)

%Restablece los valores en el archivo Simulink
"comparaciónUf" pasar el dato al archivo Simulink:
set_param (‘comparaciónUf/Constant1’, ‘value’,
‘220/sqrt(3)’);
set_param (‘comparaciónUf/Constant’, ‘value’,
‘2*pi*60’);
set_param (‘comparaciónUf/Ramp1’, ‘slope’,
‘(220/sqrt(3)-10)/0.75’);
set_param (‘comparaciónUf/Ramp’, ‘slope’,
‘120*pi/0.75‘);
set_param (‘comparaciónUf/Constant2’, ‘value’,
‘220/sqrt(3)’;
set_param (‘comparaciónUf/Ramp2’, ‘slope’,
‘120*pi/0.75’);

%Start
set_param (gcs, ‘SimulationCommand’, ‘Start’)

%Guardamos nombre función
guidata (reset_Callback);
```

La programación del botón “Cargar datos” es:

```
%Botón datos Excel: Copia los valores del archivo Excel
seleccionado a los editores y deslizadores del GUI y
tambien los pasa al sistema Simulink “ComparaciónUf”.
     Function Abrirexceldatos_Callback (hObject,~,handles)

     %Muestra un mensaje en el panel de texto
          Set (handles.text, ‘string’, ‘Ha Elegido cargar
          los datos de un archivo Excel’);

     %Se cambia la figura del puntero del ratón por un
     reloj de espera
          set (handles.figure1, ‘Pointer’, ‘watch’);

     %Abre una ventana emergente para seleccionar un
     archivo Excel y cargarlo. FileName, Path y FilterIndex
     son simples variables
          [FileName Path] = uigetfile ({‘*.xls’, ‘Archivo
          Excel (*.xls)’}, ‘Abrir Archivo Excel’,
          ‘Documentación/General (Guardar y Cargar).xls’);

     %Si le damos al botón cancelar…
          if isequal (FileName, 0)

               %Quitar el reloj de espera en el puntero
               set (handles.figure1, ‘Pointer’, ‘arrow’);

               %Retorna al GUI
               return
          else

          %Barra de espera
          %Abre o actualiza una ventana de diálogo de la
          barra de espera
               h_waitbar = waitbar (0, ‘Por favor,
               espere...’, ‘Position’, [337, 430, 270, 50],
               ‘Tag’, ‘close_waitbar’, ‘CloseRequestFcn’,
               @close_waitbar_CloseRequestFcn);

          %Edita la barra de espera 300 veces
               for i = 1:300

               %Determina si la entrada es válida
                    if ishandle (h_waitbar)

                         %Si es válida, repite la imagen e
                         incrementa la unidad (i)
                         waitbar (i / 300, h_waitbar)
```

```
                else
                %Si no es válida, retorna al GUI
                     Break
                     end
           end

           %Cierra la ventana
                delete (h_waitbar)

           %Abre el archivo seleccionado
                winopen (strcat (Path, FileName));
     end

%Guardamos el nombre del archivo seleccionado y lo
guardamos en una variable
     handles.datosexcel = FileName;
     guidata (hObject, handles);

%Restamos ".xls" a filename para que quede solo el
nombre del archivo y lo guardamos en una variable
     Datosexcel = handles.datosexcel (1, 1:length
     (handles.datosexcel)-4);

%Unimos la ruta con el nombre
     var100 = strcat ('Documentacion/', datosexcel);

%Introducimos los valores seleccionados en los edits,
en los sliders y en el archivo Simulink
     [dexcel] = xlsread (var100, 'F4:F4');

%Guardamos el valor en memoria gracias al
identificador
     handles.dexcel = dexcel;
     guidata (hObject, handles);

%Introducimos el valor contenido en el archivo Excel
en el editor deslizador
     set(handles.editC1,'string',
     num2str(handles.dexcel));

%Introducimos el valor contenido en el archivo Excel
en el deslizador
     Set (handles.sliderC1, 'Value', handles.dexcel);

%Condición de alarma
     if (handles.dexcel < 128);

     %Muestra un mensaje en el panel de texto general.
     Para borrar lo escrito en el static text y que no
     ponga que hay alarma
          set (handles.texta, 'String', ' ');
```

```
     %Deshabilita el Check box “alarma1".
          set (handles.alarma1, ‘Value’, 0);

     %Devolvemos al frame su estado de no alarma
          set(handles.frame1,       ‘Backgroundcolor’,
          ‘factory’)
     else

     %Muestra un mensaje en el panel de texto general
          set (handles.texta, ‘String’, ‘PRECAUCIÓN!!!
          SUPERA EL VALOR NOMINAL (220/sqrt(3))’);

     %Habilita el Check box "alarma1".
          set (handles.alarma1, ‘Value’, 1);

     %El frame cambia ha estado de alarma
          set    (handles.frame1,    ‘Backgroundcolor’,
          ‘red’)
     end

%Se guardan los identificadores
     guidata (hObject, handles);

%Se pasa el dato al archivo Simulink:
     set_param  (‘comparaciónUf/Constant1’,   ‘value’,
     num2str (handles.dexcel));

%Introducimos los valores seleccionados en los edits,
en los sliders y en el archivo Simulink
     [dexcel] = xlsread (var100, ‘F5:F5’);

%Se guarda el valor en memoria gracias alidentificador
     handles.dexcel = dexcel;
     guidata (hObject, handles);

%Pone el valor en el editor
     set (handles.editC2, ‘string’, num2str
     (handles.dexcel));

%Pone el valor en el deslizador
     set (handles.sliderC2, ‘Value’, handles.dexcel);

%Condición de alarma
     if (handles.dexcel < 314);

%Muestra un mensaje en el panel de texto general
     set (handles.textb, ‘String’, ‘PRECAUCIÓN!!! La
     frecuencia es menor de 50 Hz’);
```

```
%Habilita el Check box alarma1
     set (handles.alarma2, ‘Value’, 1);

%El frame cambia ha estado de alarma
     set (handles.frame2, ‘Backgroundcolor’, ‘red’)
     elseif (handles.dexcel >377);

%Muestra un mensaje en el panel de texto general
     set (handles.textb, ‘String’, ‘PRECAUCIÓN!!! La
     frecuencia es mayor de 60 Hz’);

%Habilita el Check box alarma2
     set (handles.alarma2, ‘Value’, 1);

%El frame cambia ha estado de alarma
     set (handles.frame2, ‘Backgroundcolor’, ‘red’)
     else

%Muestra un mensaje en el panel de texto general
     set (handles.textb, ‘String’, ‘ ’);

%Para borrar lo escrito en el static text y que no
ponga que hay alarma
%Deshabilita el Check box "alarma2".
     set (handles.alarma2, ‘Value’, 0);

%El frame cambia ha estado de alarma
     set(handles.frame2, ‘Backgroundcolor’, ‘factory’)
     end

%Se guardan los identificadores
     guidata (hObject, handles);

%Para pasar el dato al archivo Simulink:
     set_param   (‘comparaciónUf/Constant’,   ‘value’,
     num2str (handles.dexcel));

%Guardamos los datos del archivo Excel en una variable
     [dexcel] = xlsread (var100, ‘G4:G4’);

%Guardamos el valor en memoria gracias al
identificador
     handles.dexcel = dexcel;
     guidata (hObject, handles);

%Introducimos el valor contenido en el archivo Excel
en el editor
     set (handles.editR1, ‘string’, num2str
     (handles.dexcel));
```

```
%Introducimos el valor contenido en el archivo Excel
en el deslizador
     set (handles.sliderR1, 'Value', handles.dexcel);

%Condición de alarma
     if (handles.dexcel < 128);

%Muestra un mensaje en el panel de texto general
     set (handles.textc, 'String', ' ');

%Para borrar lo escrito en el static text y que no
ponga que hay alarma

%Deshabilita el Check box "alarma3".
     set (handles.alarma3, 'Value', 0);

%El frame cambia ha estado de alarma
     set      (handles.frame3,      'Backgroundcolor',
     'factory')
     else

%Muestra un mensaje en el panel de texto general
     set   (handles.textc,   'String',   'PRECAUCIÓN!!!
     SUPERA EL VALOR NOMINAL (220/sqrt(3))');

%Habilita el Check box alarma1
     set (handles.alarma3, 'Value', 1);

%El frame cambia ha estado de alarma
     set (handles.frame3, 'Backgroundcolor', 'red')
     end

%Se guardan los identificadores
     guidata (hObject, handles);

%Para pasar el dato al archivo Simulink:
     set_param    ('comparaciónUf/Ramp1',    'slope',
     num2str (handles.dexcel));

%Introducimos los valores seleccionados en los edits,
en los sliders y en el archivo Simulink
     [dexcel] = xlsread (var100, 'G5:G5');

%Guardamos el valor en memoria gracias al
identificador
     handles.dexcel = dexcel;
     guidata (hObject, handles);

%Introducimos el valor contenido en el archivo Excel
en el editor
```

```
     set (handles.editR2, ‘string’, num2str
     (handles.dexcel));

%Introducimos el valor contenido en el archivo Excel
en el deslizador
     set (handles.sliderR2, ‘Value’, handles.dexcel);

%Condición de alarma
     if (handles.dexcel < 314);

%Muestra un mensaje en el panel de texto general
     set (handles.textd, ‘String’, ‘PRECAUCIÓN!!! La
     frecuencia es menor de 50 Hz’);

%Habilita el Check box alarma4
     set (handles.alarma4, ‘Value’, 1);

%El frame cambia ha estado de alarma
     set (handles.frame4, ‘Backgroundcolor’, ‘red’)

     elseif (handles.dexcel > 377);

%Muestra un mensaje en el panel de texto general
     set (handles.textd, ‘String’, ‘PRECAUCIÓN!!! La
     frecuencia es mayor de 60 Hz’);

%Habilita el Check box alarma4
     set (handles.alarma4, ‘Value’, 1);

%El frame cambia ha estado de alarma
     set (handles.frame4, ‘Backgroundcolor’, ‘red’)
      else

%Muestra un mensaje en el panel de texto general
      set (handles.textd, ‘String’, ‘ ’);

%Para borrar lo escrito en el static text y que no
ponga que hay alarma

%Deshabilita el Check box "alarma4".
     set (handles.alarma4, ‘Value’, 0);

%El frame cambia ha estado de alarma
     set(handles.frame4, ‘Backgroundcolor’, ‘factory’)
     end

%Se guardan los identificadores
     guidata (hObject, handles);

%Para pasar el dato al archivo Simulink:
```

```
     set_param (‘comparaciónUf/Ramp’, ‘slope’, num2str
     (handles.dexcel));

%introducimos los valores seleccionados en los edits,
en los sliders y en el archivo Simulink
     [dexcel] = xlsread (var100, ‘H4:H4’);

%Guardamos   el   valor   en   memoria   gracias   al
identificador
     handles.dexcel = dexcel;
     guidata (hObject, handles);

%Introducimos el valor contenido en el archivo Excel
en el editor
     set (handles.editCR1, ‘string’, num2str
     (handles.dexcel));

%Introducimos el valor contenido en el archivo Excel
en el deslizador
     set (handles.sliderCR1, ‘Value’, handles.dexcel);

%Condición de alarma
     if (handles.dexcel < 128);

%Muestra un mensaje en el panel de texto general
     set (handles.texte, ‘String’, ‘ ’);

%para  borrar  lo  escrito  en  el  static  text  y  que  no
ponga que hay alarma

%Deshabilita el Check box "alarma5".
     set (handles.alarma5, ‘Value’, 0);

%El frame cambia ha estado de alarma
     set      (handles.frame5,      ‘Backgroundcolor’,
     ‘factory’)
     else

%Muestra un mensaje en el panel de texto general
     set   (handles.texte,   ‘String’,   ‘PRECAUCIÓN!!!
     SUPERA EL VALOR NOMINAL (220/sqrt(3))’);

%Habilita el Check box alarma5
     set (handles.alarma5, ‘Value’, 1);

%El frame cambia ha estado de alarma
     set (handles.frame5, ‘Backgroundcolor’, ‘red’)
     end

%Se guardan los identificadores
     guidata (hObject, handles);
```

```
%Para pasar el dato al archivo Simulink:
     set_param  (‘comparaciónUf/Constant2’,  ‘value’,
     num2str (handles.dexcel));

%Introducimos los valores seleccionados en los edits,
en los sliders y en el archivo Simulink
     [dexcel] = xlsread (var100, ‘H5:H5’);

%Guardamos el valor en memoria gracias al
identificador
     handles.dexcel = dexcel;
     guidata (hObject, handles);

%Introducimos el valor contenido en el archivo Excel
en el editor
     set (handles.editCR2, ‘string’, num2str
     (handles.dexcel));

%Introducimos el valor contenido en el archivo Excel
en el deslizador
     set (handles.sliderCR2, ‘Value’, handles.dexcel);

%Condición de alarma
     if (handles.dexcel < 314);
     %Muestra un mensaje en el panel de texto general
          set (handles.textf, ‘String’, ‘PRECAUCIÓN!!!
          La frecuencia es menor de 50 Hz’);

     %Habilita el Check box alarma6
          set (handles.alarma6, ‘Value  , 1);

     %El frame cambia ha estado de alarma
          set   (handles.frame6,   ‘Backgroundcolor’,
          ‘red’)

     elseif (handles.dexcel > 377);

     %Muestra un mensaje en el panel de texto general
          set (handles.textf, ‘String’, ‘PRECAUCIÓN!!!
          La frecuencia es mayor de 60 Hz’);

     %Habilita el Check box alarma6
          set (handles.alarma6, ‘Value’, 1);

     %El frame cambia ha estado de alarma
          set   (handles.frame6,   ‘Backgroundcolor’,
          ‘red’)

     else
```

```
        %Muestra un mensaje en el panel de texto general
             set (handles.textf, 'String', ' ');

        %Para borrar lo escrito en el static text y que
        no ponga que hay alarma

        %Deshabilita el Check box "alarma6".
             set (handles.alarma6, ' Value', 0);

        %El frame cambia ha estado de alarma
             set(handles.frame6,'Backgroundcolor','factor
             y')
        end

   %Se guardan los identificadores
        guidata(hObject,handles);

   %Para pasar el dato al archivo Simulink:
        set_param('comparaciónUf/Ramp2', 'slope', num2str
        (handles.dexcel));

   %Quitar el reloj de espera en el puntero
        set (handles.figure1, 'Pointer', 'arrow');

   %Guarda el nombre de la función
        guidata (Abrirexceldatos_Callback);
```

Y por último está el botón info.

Similar a todos los botones info. anteriores.

5.4 EXPLICACIÓN DEL PANEL DE CONTROL DCHA. [4]

En la parte derecha, vemos un panel de control manejado por sus botones o pestañas (Marcadas con la flecha en rojo), tal como se muestra en la figura 72:

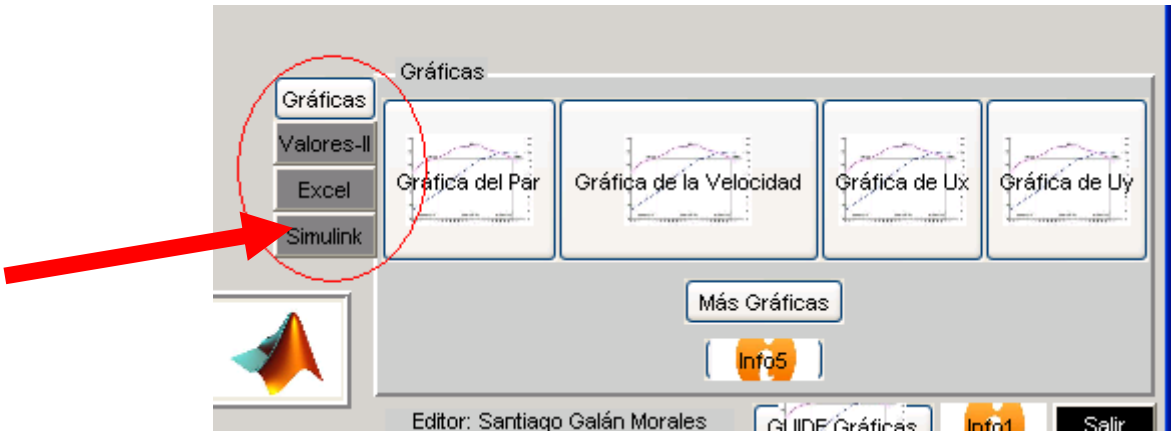

Figura 72. Pestañas

Esta parte de nuestro GUI, también es muy importante ya que aquí cae gran parte de trabajo.

En esta parte tenemos cuatro paneles de control, de los cuales, tres están ocultos y uno está visible. Los que hacen que panel de control es el visible, son los botones o pestañas señalados por la flecha roja. El seleccionado será el que muestre su panel de control respectivo. La pestaña “Gráficas” se refiere al panel de control “gráficas”. La pestaña “Valores-II” se refiere al panel de control “Muestra valores Max, Min y medios”. La pestaña “Excel” se refiere al panel de control “Guarda valores en Excel”, La pestaña “Simulink” se refiere al panel de control “guarda archivo Simulink”. Todos estos paneles de control se refieren al archivo Simulink “compardcionUf”.

En adelante explicaremos los cuatro paneles de control.

Panel de control: “Gráficas”, mostrado en la figura 73.

Figura 73. Panel de control dcha.

Este panel de control, nos permite graficar las diferentes señales que el archivo Simulink “ComparaciónUf” nos da. De este archivo podemos sacar las señales del par, de la velocidad y de las tensiones de una máquina eléctrica. Por tanto el panel de control, ilustra cuatro botones, los cuales se reparten las señales. Por ejemplo si pulsamos el botón “Par” veremos las señales que se muestran en la figura 74:

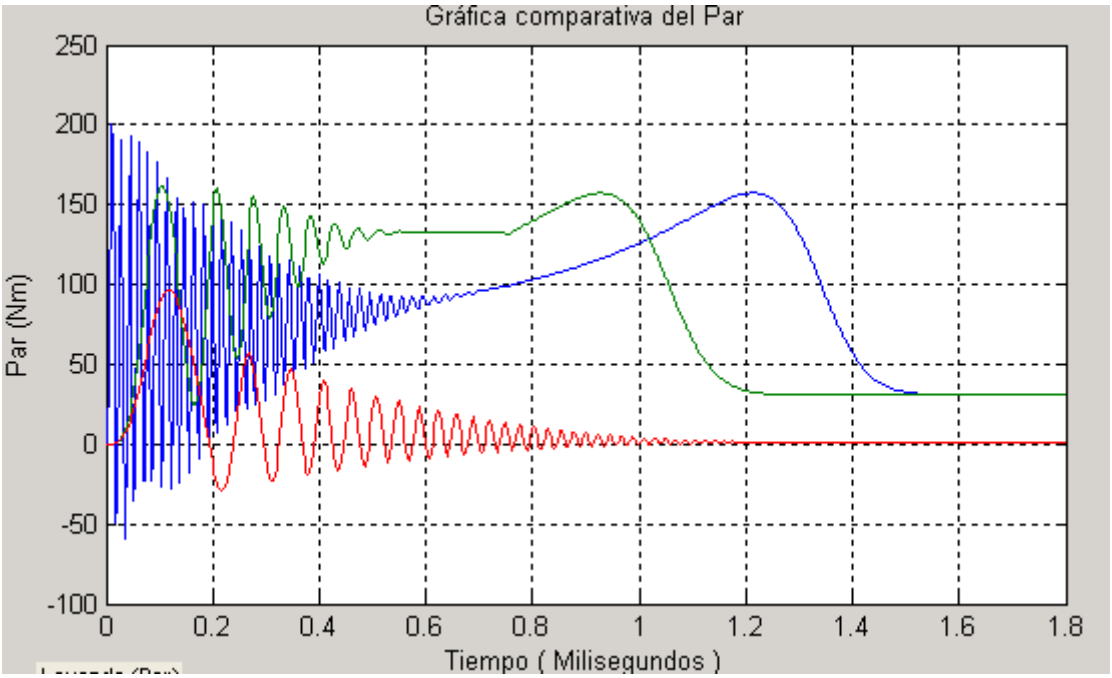

Figura 74. Gráficas

Ajustándose a los valores introducidos anteriormente.

La imagen de este botón se muestra en la figura 75:

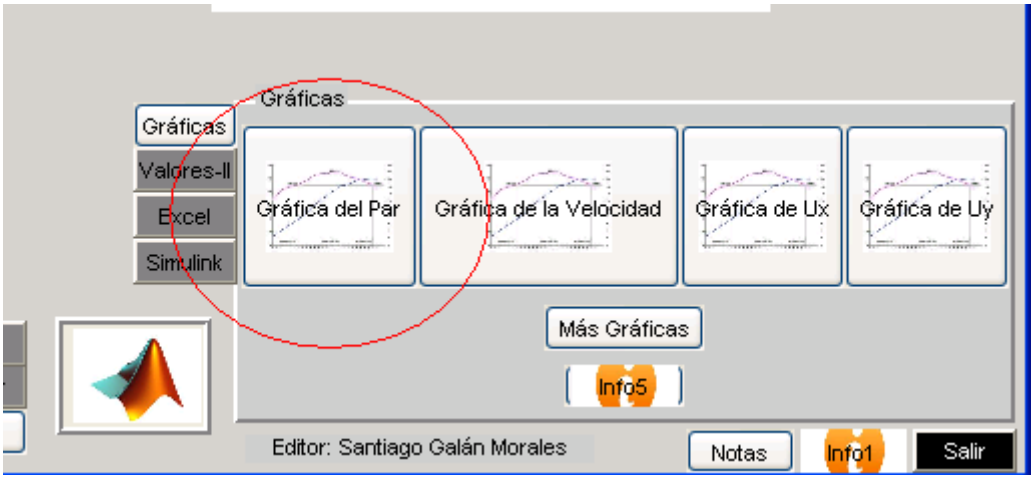

Figura 75. Botón Gráfica del Par.

La programación de este botón es:

```
%Función senalesp: Esta función hace referencia al botón
"señalesp" que nos muestra en el panel de control gráfico
la gráfica comparativa del par
     function senalesp_Callback (hObject, ~, handles)

     %Se cambia la figura del puntero del ratón por un
     reloj de espera
          set (handles.figure1, 'Pointer', 'watch');

     %Inhabilita los botones para graficar
          set (handles.senalesp, 'Enable', 'off');
          set (handles.senalesv, 'Enable', 'off');
          set (handles.senalesUx, 'Enable', 'off');
          set (handles.senalesUy, 'Enable', 'off');

     %Hacemos aparecer el panel de control Leyenda 1, el
     resto los ocultamos por si alguno estuviera al
     descubierto
          set (handles.PanelLeyenda1, 'Visible', 'on');
          set (handles.PanelLeyenda2, 'Visible', 'off');
          set (handles.PanelLeyenda3, 'Visible', 'off');
          set (handles.PanelLeyenda4, 'Visible', 'off');

     %Seleccionamos los tres "radiobutton" del panel de
     control de leyenda desde el principio
          set (handles.RadioButtonCtePar, 'Value', 1);
          set (handles.RadioButtonRampaPar, 'Value', 1);
          set (handles.RadioButtonCteRamPar, 'Value', 1);
```

```
%Variables que nos dicen si las señales están
activadas o no, en este caso hemos seleccionado las
tres
     handles.varselctepar = 1;
     handles.varselrampar = 1;
     handles.varselcterampar = 1;
     guidata (hObject, handles);

%Barra de espera
%Abre o actualiza una ventana de diálogo de la barra
de espera
     h_waitbar = waitbar (0, 'Por favor, espere...',
     'Position', [337, 430, 270, 50], 'Tag',
     'close_waitbar', 'CloseRequestFcn',
     @close_waitbar_CloseRequestFcn);

%Edita la barra de espera 300 veces
     for i = 1:300

     %Determina si la entrada es válida
          if ishandle (h_waitbar)

               %Si es válida, repite la imagen e
               incrementa la unidad (i)
               waitbar (i / 300, h_waitbar)

          else

               %Si no es válida, retorna al GUI
               break

          end

     end

%Cierra la ventana
     delete (h_waitbar)
     if handles.valcond == 0

     %Verifica que existe el archivo Simulink
     ComparaciónUf
     find_system('Name','ArchivosSimulink\comparaciónU
     f');

     %Abrir del menú archivo Simulink comparaciónuf
     open_system ('ArchivosSimulink\comparaciónUf');

     %Condición para que el archivo comparaciónUf no
     vuelva a abrirse
     valcond = 1;
     handles.valcond = valcond;
```

```
     guidata (hObject, handles);

     %Condición para que se cierre el archivo Simulink
     comparaciónUf
     handles.FileName = 'comparaciónUf.mdl';
     guidata (hObject, handles);
     handles.ruta = 'comparaciónUf';
     guidata (hObject, handles);
     end

%Muestra un mensaje en el panel de texto general
     set (handles.text, 'String', 'Se graficará las
     señales del Par y se mostrará los valores
     máximos, mínimos y medios en el panel de control
     "valores"');

%Designamos el axes donde irán las gráficas
     axes (handles.axes)

%Modifica el archivo Simulink
     options = simset ('SrcWorkspace', 'current');

%simula el sistema dinámico del archivo Simulink
     sim ('comparaciónUf',[],options);

%Gráfica los valores %Simula el sistema dinámico del
archivo Simulink
     plot (tout, dubai)

%Salida y variable del to workspace
%Designamos las propiedades de la gráfica
     xlabel ('Tiempo(Milisegundos)')

%título ejes
     Ylabel ('Par (Nm)')

%título ejes
     Title ('Gráfica comparativa del Par')

%título axes
     grid on %cuadrícula

%Valores máx., min. y medios para el par máx.
     max1 = max (dubai4);
     handles.max1 = max1;

%Muestra un mensaje en el panel de control de Valores-
II
     Set(handles.text11,'String',num2str(handles.max1)
     );
     guidata (hObject, handles);
```

```
     max2 = max (dubai5);
     handles.max2 = max2;

%Muestra un mensaje en el panel de control de Valores-
II
     Set(handles.text21,'String',num2str(handles.max2)
     );
     Guidata (hObject, handles);
     max3 = max (dubai6);
     handles.max3 = max3;

%Muestra un mensaje en el panel de control de Valores-
II
     set (handles.text31, 'String', num2str
     (handles.max3));
     guidata (hObject, handles);

%min
     min1 = min (dubai4);
     handles.min1 = min1;

%Muestra un mensaje en el panel de control de Valores-
II
     Set(handles.text12,'String',num2str(handles.min1)
     );
     guidata (hObject, handles);
     min2 = min (dubai5);
     handles.min2 = min2;

%Muestra un mensaje en el panel de control de Valores-
II
     Set(handles.text22,'String',num2str(handles.min2)
     );
     guidata(hObject,handles);
     min3 = min (dubai6);
     handles.min3 = min3;

%Muestra un mensaje en el panel de control de Valores-
II
     Set(handles.text32,'String',num2str(handles.min3)
     );
     Guidata (hObject, handles);

%mean
     mean1 = mean (dubai4);
     handles.mean1 = mean1;

%Muestra un mensaje en el panel de control de Valores-
II
     Set(handles.text13,'String',num2str(handles.mean1
     ));
```

```
     Guidata (hObject, handles);
     mean2 = mean (dubai5);
     handles.mean2 = mean2;

%Muestra un mensaje en el panel de control de Valores-
II
     Set(handles.text23,'String',num2str(handles.mean2
     ));
     Guidata (hObject,handles);
     mean3 = mean (dubai6);
     handles.mean3 = mean3;

%Muestra un mensaje en el panel de control de Valores-
II
     Set(handles.text33,'String',num2str(handles.mean3
     ));
     guidata (hObject, handles);

%Valores máx., min. y medios para velocidades
%max
     max12 = max (dubai4);
     handles.max12 = max12;
     guidata (hObject, handles);
     max22 = max (dubai8);
     handles.max22 = max22;
     guidata (hObject, handles);
     max32 = max (dubai9);
     handles.max32 = max32;
     guidata (hObject, handles);

%min.
     min12 = min (dubai7);
     handles.min12 = min12;
     guidata (hObject, handles);
     min22 = min (dubai8);
     handles.min22 = min22;
     guidata (hObject, handles);
     min32 = min (dubai9);
     handles.min32 = min32;
     guidata (hObject, handles);

%mean
     mean12 = mean (dubai7);
     handles.mean12 = mean12;
     guidata (hObject, handles);
     mean22 = mean (dubai8);
     handles.mean22 = mean22;
     guidata (hObject,handles);
     mean32 = mean (dubai9);
     handles.mean32 = mean32;
     guidata (hObject, handles);
```

```
%valores máx., min y medios para Ux
%máx.
     max13 = max (dubai10);
     handles.max13 = max13;
     guidata (hObject, handles);
     max23 = max (dubai11);
     handles.max23 = max23;
     guidata (hObject, handles);
     max33 = max (dubai12);
     handles.max33 = max33;
     guidata (hObject, handles);

%min
     min13 = min (dubai10);
     handles.min13 = min13;
     guidata (hObject, handles);
     min23 = min (dubai11);
     handles.min23 = min23;
     guidata (hObject, handles);
     min33 = min (dubai12);
     handles.min33 = min33;
     guidata (hObject, handles);

%mean
     mean13 = mean (dubai10);
     handles.mean13 = mean13;
     guidata (hObject, handles);
     mean23 = mean (dubai11);
     handles.mean23 = mean23;
     guidata (hObject, handles);
     mean33 = mean (dubai12);
     handles.mean33 = mean33;
     guidata (hObject, handles);

%Valores máx., min y medios para Uy
%max
     max14 = max (dubai13);
     handles.max14 = max14;
     guidata (hObject, handles);
     max24 = max (dubai14);
     handles.max24 = max24;
     guidata (hObject, handles);
     max34 = max (dubai15);
     handles.max34 = max34;
     guidata (hObject, handles);

%min
     min14 = min (dubai13);
     handles.min14 = min14;
     guidata (hObject, handles);
```

```
        min24 = min (dubai14);
        handles.min24 = min24;
        guidata (hObject, handles);
        min34 = min(dubai15);
        handles.min34 = min34;
        guidata (hObject, handles);

    %mean
        mean14 = mean (dubai13);
        handles.mean14 = mean14;
        guidata (hObject, handles);
        mean24 = mean (dubai14);
        handles.mean24 = mean24;
        guidata (hObject, handles);
        mean34 = mean (dubai15);
        handles.mean34 = mean34;
        guidata (hObject, handles);

    %Mensajes a los editores correspondientes a los
    valores máx., min y mean
        set (handles.text52, 'String', 'Valores Max del
        PAR');
        set (handles.text53, 'String', 'Valores Min del
        PAR');
        set(handles.text54,'String','Valores Medios del
        PAR');

    %Los botones vuelven a estar activos y el puntero a
    ser una flecha
        set (handles.figure1, 'Pointer', 'arrow');

    %Habilita los botones
        set (handles.senalesp, 'Enable', 'on');
        set (handles.senalesv, 'Enable', 'on');
        set (handles.senalesUx, 'Enable', 'on');
        set (handles.senalesUy, 'Enable', 'on');

    %Guarda el nombre de la función
        guidata (senalesp_Callback);
```

El resto de botones para graficar son similares.

Como son botones de comparación entre señales, en el panel de graficar, vemos tres de ellas, cada una del par de cada uno de los tres circuitos por los que está compuesto el archivo Simulink "ComparaciónUf ", si lo que queremos es visualizar una o dos señales indistintamente, tenemos un panel de control con tres "Leyenda", que hasta este momento había estado oculto por que no nos hacía falta, los cuales controlan las tres señales. Cada botón comparativo, tienen su propio panel leyenda propio, tal y

como se muestra en la figura 76, que irá apareciendo según se pida su respectiva gráfica.

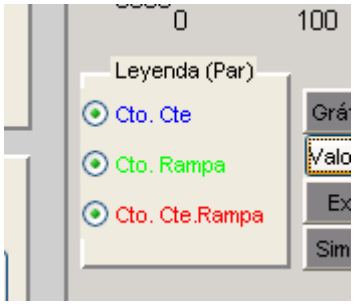

Figura 76. Leyenda.

La programación de uno de los “Radio-Button” de un solo panel Leyenda es:

-Programación del “Radio-Button Cto. Cte” del panel leyenda (Par):

```
%Función RadioButtonCtePar: Esta función hace referencia al
radiobutton "RadioButtonCtePar" del panel de leyenda del
par. Una vez seleccionado las señales a mostrar se hacen
aparecer en la gráfica
     function   RadioButtonCtePar_Callback   (hObject,   ~,
     handles)

     %Ejecuta la acción del radiobutton
          if (get(hObject, ‘Value’) == get(hObject, ‘Max’))

               %Si   se   ha   habilitado   el   radiobutton
               igualamos la variable a uno
               handles.varselctepar = 1;
               guidata (hObject, handles);
          else

               %Si   se   ha   deshabilitado   el   radiobutton
               igualamos la variable a cero
               handles.varselctepar = 0;
               guidata (hObject, handles);
          end

     %Según como estén igualadas las variables, estas se
     graficaran o no
          if (handles.varselctepar == 0) &&
          (handles.varselrampar == 0) &&
          (handles.varselcterampar == 0)

     %Muestra un mensaje en el panel de texto general
          set(handles.text,‘string’,‘No   ha   seleccionado
          ninguna gráfica’)

     %No gráfica nada
```

```
     plot (0)
     elseif (handles.varselctepar==0) &&
     (handles.varselrampar==0) &&
     (handles.varselcterampar==1)


%Muestra un mensaje en el panel de texto general
     set(handles.text,‘string’,‘Ha   seleccionado   la
     gráfica del par del circuito Cte.Rampa’)

%Designamos el axes donde irán las gráficas
     axes (handles.axes)

%Modifica el archivo Simulink
     Options = simset (‘SrcWorkspace’, ‘current’);

%Simula el sistema dinámico del archivo Simulink
     sim (‘comparaciónUf’, [ ], options);

%Nombre archivo Simulink, variable
%Gráfica los valores %Simula el sistema dinámico del
archivo Simulink
     Plot (tout, dubai6, ‘color’, ‘red’)

%Designamos las propiedades de la gráfica
     xlabel (‘Tiempo(Milisegundos)’)

%título ejes
     ylabel(‘Par(Nm)’)

%título ejes
     title(‘Gráfica comparativa del Par’)

%título axes
     grid on

%cuadrícula
     elseif (handles.varselctepar==0) &&
     (handles.varselrampar==1) &&
     (handles.varselcterampar==0)

%Muestra un mensaje en el panel de texto general
     Set (handles.text, ‘string’, ‘Ha seleccionado la
     gráfica del par del circuito Rampa’)

%Designamos el axes donde irán las gráficas
     axes (handles.axes)

%Modifica el archivo Simulink
     Options = simset (‘SrcWorkspace’, ‘current’);
```

```
%Simula el sistema dinámico del archivo Simulink
Sim (‘comparaciónUf’, [], options);

%Gráfica los valores, simula el sistema dinámico del
archivo Simulink
%Designamos las propiedades de la gráfica
     Plot (tout, dubai5, ‘color’, ‘green’)

%Se designa los títulos a los ejes
     Xlabel (‘Tiempo (Milisegundos)’)
     Ylabel (‘Par (Nm)’)

%Se designa el título de la gráfica.
     Title (‘Gráfica comparativa del Par’)

%Cuadrícula
     grid on

     elseif (handles.varselctepar==0) &&
     (handles.varselrampar==1) &&
     (handles.varselcterampar==1)

%Muestra un mensaje en el panel de texto general
     set (handles.text, ‘string’, ‘Ha seleccionado las
     gráficas del par del circuito Rampa y del
     circuito Cte.Rampa’)

%Designamos el axes donde irán las gráficas
     axes (handles.axes)

%Modifica el archivo Simulink
     options = simset (‘SrcWorkspace’, ‘current’);

%Simula el sistema dinámico del archivo Simulink
     sim (‘comparaciónUf’, [], options);

%Gráfica los valores %Simula el sistema dinámico del
archivo Simulink
%Se designan las propiedades de la gráfica
     plot (tout, dubai18)

%Se designa los títulos a los ejes
     xlabel (‘Tiempo(Milisegundos)’)
     ylabel(‘Par(Nm)’)

%Se designa el título de la gráfica
     title(‘Gráfica comparativa del Par’)

%Cuadrícula
     grid on
```

```
     elseif (handles.varselctepar==1) &&
     (handles.varselrampar==0) &&
     (handles.varselcterampar==0)

%Muestra un mensaje en el panel de texto general
     set (handles.text, 'string', 'Ha seleccionado la
     gráfica del par del circuito Cte.')

%Designamos el axes donde irán las gráficas
     axes (handles.axes)

%Modifica el archivo Simulink
     options = simset ('SrcWorkspace', 'current');

%Simula el sistema dinámico del archivo Simulink
     sim('comparaciónUf',[],options);

%Gráfica los valores %Simula el sistema dinámico del
archivo Simulink
%Designamos las propiedades de la gráfica
     plot(tout,dubai4)

%Se designa los títulos a los ejes
     xlabel ('Tiempo(Milisegundos)')
     ylabel('Par(Nm)')

%Se designa el título de la gráfica
     title('Gráfica comparativa del Par')

%Cuadrícula
     grid on

     elseif (handles.varselctepar==1) &&
     (handles.varselrampar==0) &&
     (handles.varselcterampar==1)

%Muestra un mensaje en el panel de texto general
     set (handles.text, 'string', 'Ha seleccionado las
     gráficas del par del circuito Cte. y del circuito
     Cte.Rampa')

%Designamos el axes donde irán las gráficas
     axes (handles.axes)

%Modifica el archivo Simulink
     options = simset ('SrcWorkspace', 'current');

%Simula el sistema dinámico del archivo Simulink
     sim('comparaciónUf',[],options);
```

```
%Gráfica los valores %Simula el sistema dinámico del
archivo Simulink
     plot (tout, dubai17)

%Designamos las propiedades de la gráfica
     plot(tout,dubai4)

%Se designa los títulos a los ejes
     xlabel ('Tiempo(Milisegundos)')
     ylabel('Par(Nm)')

%Se designa el título de la gráfica
     title('Gráfica comparativa del Par')

%Cuadrícula
     grid on

     elseif (handles.varselctepar==1) &&
     (handles.varselrampar==1) &&
     (handles.varselcterampar==0)

%Muestra un mensaje en el panel de texto general
     set (handles.text, 'string', 'Ha seleccionado las
     gráficas del par del circuito Cte y del circuito
     Rampa')

%Designamos el axes donde irán las gráficas
axes (handles.axes)

%Modifica el archivo Simulink
     options = simset ('SrcWorkspace', 'current');

%Simula el sistema dinámico del archivo Simulink
     sim('comparaciónUf', [], options);

%Gráfica los valores %Simula el sistema dinámico del
archivo Simulink
%Designamos las propiedades de la gráfica
     plot(tout, dubai16)

%Se designa los títulos a los ejes
     xlabel ('Tiempo(Milisegundos)')
     ylabel ('Par(Nm)')

%Se designa el título de la gráfica
     title('Gráfica comparativa del Par')

%Cuadrícula
     grid on
```

```
     elseif (handles.varselctepar ==1) &&
     (handles.varselrampar==1) &&
     (handles.varselcterampar==1)

%Muestra un mensaje en el panel de texto general
     set (handles.text, 'string', 'Ha seleccionado las
     gráficas del par del circuito Cte., del circuito
     Rampa y del circuito Cte.Rampa·')

%Designamos el axes donde irán las gráficas
     axes (handles.axes)

%Modifica el archivo Simulink
     options = simset ('SrcWorkspace', 'current');

%Simula el sistema dinámico del archivo Simulink
     sim ('comparaciónUf', [], options);

%Nombre archivo Simulink, variable
%Gráfica los valores %Simula el sistema dinámico del
archivo Simulink
%Designamos las propiedades de la gráfica
     plot (tout, dubai)

%Se designa los títulos a los ejes
     xlabel ('Tiempo(Milisegundos)')
     ylabel ('Par(Nm)')

%Se designa el título de la gráfica
     title('Gráfica comparativa del Par')

%Cuadrícula
     grid on
     else

%Muestra un mensaje en el panel de texto general
     Set (handles.text, 'string', 'error')
     end

%Guarda el nombre de la función
     guidata (RadioButtonCtePar_Callback);
```

También nos encontramos el botón "info" el cual nos da información sobre este panel.Además, nos encontramos un botón llamado "Más gráficas", tal y como se muestra en la figura 77:

Figura 77. Botón Gráficas

Este elemento nos lleva a un quinto panel de control llamado “Más gráficas”, que nos proporciona más botones que nos dan la posibilidad de graficar más señales, tal y como se muestra en la figura 78.

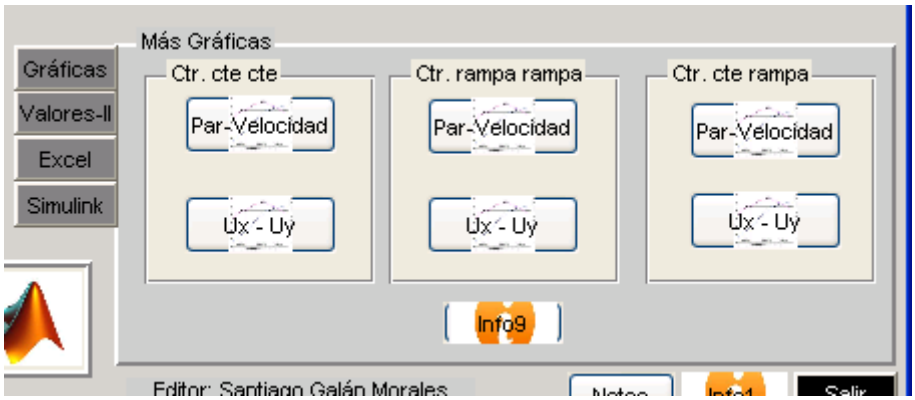

Figura 78. Más Gráficas

El botón Ux-Uy del circuito Rampa-Rampa, dependiendo de los valores introducidos, nos muestra la señal, tal y como se muestra en la figura 79:

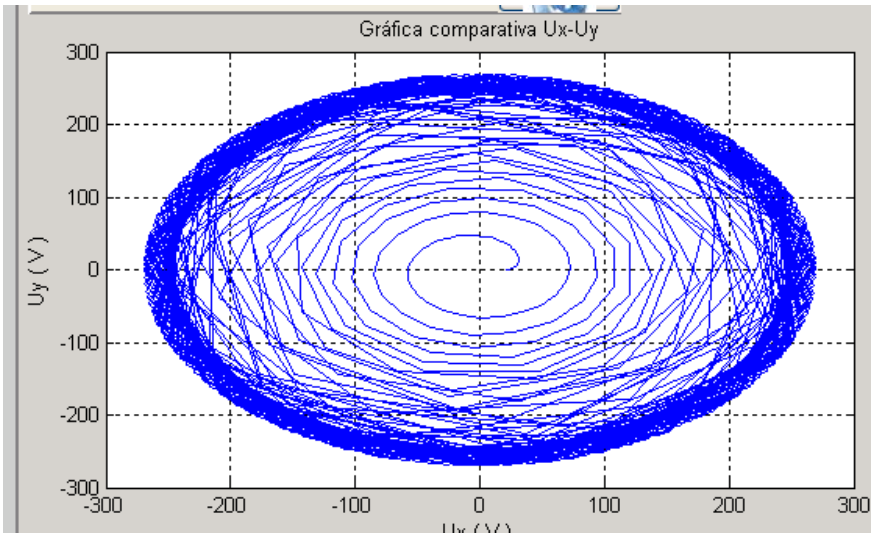

Figura 79. Grafica la señal Ux-Uy.

La programación de este elemento es:

```
%Función bUxUy2: Esta función hace referencia al botón
"bUxUy2" que nos muestra en el panel de control gráfico la
gráfica comparativa entre la tensión Ux y Uy del circuito
cte rampa
     function bUxUy2_Callback (hObject, ~, handles)

     %Muestra un mensaje en el panel de texto general
          Set (handles.text, 'string', 'Ha seleccionado la
          gráfica par-velocidad del circuito rampa-rampa')

     %Se cambia la figura del puntero del ratón por un
     reloj de espera
          set (handles.figure1, 'Pointer', 'watch');

     %Desactivo los botones
          set (handles.bParVel1, 'Enable', 'off');
          set (handles.bParVel2, 'Enable', 'off');
          set (handles.bParVel3, 'Enable', 'off');
          set (handles.bUxUy1, 'Enable', 'off');
          set (handles.bUxUy2, 'Enable', 'off');
          set (handles.bUxUy3, 'Enable', 'off');

     %Barra de espera
     %Abre o actualiza una ventana de diálogo de la barra
     de espera
          h_waitbar = waitbar (0, 'Por favor, espere...',
          'Position',[337, 430, 270, 50], 'Tag',
          'close_waitbar', 'CloseRequestFcn',
          @close_waitbar_CloseRequestFcn);

     %Edita la barra de espera 300 veces
          for i = 1:300

               %Determina si la entrada es válida
               if ishandle (h_waitbar)
                    %Si es válida, repite la imagen e
                    incrementa la unidad(i)
                    waitbar (i / 300, h_waitbar)
               else
                    %Si no es válida, retorna al GUI
                    break
               end
          end

     %Cierra la ventana
          delete (h_waitbar)
          if handles.valcond == 0
```

```
%Verificar que existe el archivo Simulink
comparaciónUf
     find_system('Name','ArchivosSimulink\comparaciónU
     f');

%Abrir del menú archivo Simulink comparaciónuf
     open_system ('ArchivosSimulink\comparaciónUf');

%Condición para que el archivo comparaciónUf no vuelva
a abrirse
     valcond = 1;
     handles.valcond = valcond;
     guidata (hObject, handles);

%Condición para quese cierre el archivo Simulink
comparaciónUf
     handles.FileName = 'comparaciónUf.mdl';
     guidata (hObject, handles);
     handles.ruta = 'comparaciónUf';
     guidata (hObject, handles);
     end

%Designamos el axes donde irán las gráficas
     axes (handles.axes)

%Gráfica los valores
     Options = simset ('SrcWorkspace', 'current');

%Simula el sistema dinámico del archivo Simulink
     sim ('comparaciónUf', [], options);

%Gráfica los valores %Simula el sistema dinámico del
archivo Simulink
%Designamos las propiedades de la gráfica
     plot (dubai11, dubai14)

%Designamos los títulos de los ejes
     xlabel ('Ux(V)')
     ylabel ('Uy(V)')

%Designamos el título de la gráfica
     title ('Gráfica comparativa Ux-Uy')

%Cuadrícula
     grid on

%Activo los botones
     set (handles.bParVel1, 'Enable', 'on');
     set (handles.bParVel2, 'Enable', 'on');
     set (handles.bParVel3, 'Enable', 'on');
     set (handles.bUxUy1, 'Enable', 'on');
```

```
        set (handles.bUxUy2, 'Enable', 'on');
        set (handles.bUxUy3, 'Enable', 'on');

    %Cuando se presiona el botón este se queda inactivo y
    la flecha del cursor se vuelve relojito de espera
    hasta el final que vuelve a ser como al principio
        set (handles.figure1, 'Pointer', 'arrow');

    %Guarda el nombre de la función
        guidata (bUxUy2_Callback);
```

El resto de botones son similares.

También nos encontramos el botón "info" el cual nos da información sobre este panel.

Panel de control: "Valores – II".

Este panel nos muestra los datos que podemos tomar del archivo Simulink "ComparaciónUf ", una vez hallamos trabajado con él. Este panel nos ofrece los valores máximos, mínimos y medios de las distintas señales graficadas.

Pero solo mostrará los valores de la última señal pedida por el usuario, tal y como se muestra en la figura 80.

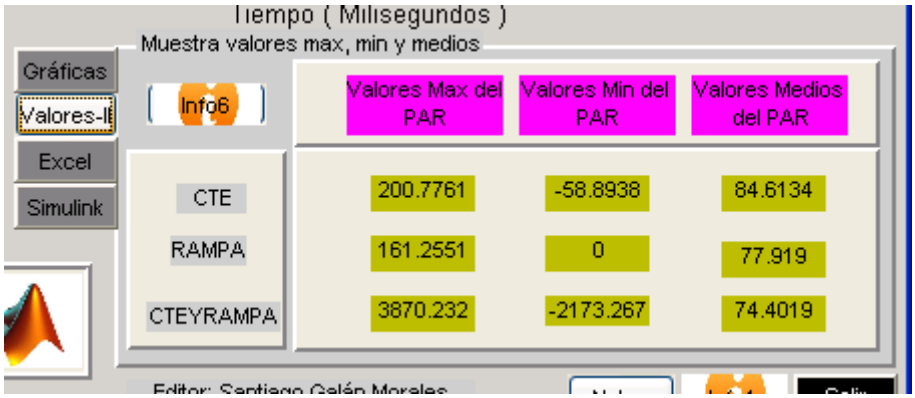

Figura 80. Valores

También nos encontramos el botón "info" el cual nos da información sobre este panel.

Panel de control, “Excel”, mostrado en la figura 81.

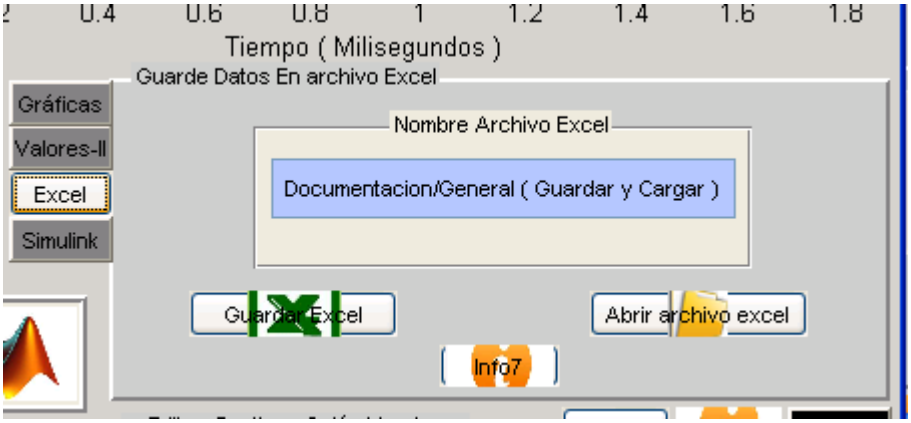

Figura 81. Excel

Este panel de control tiene la capacidad de guardar los datos introducidos y los datos adquiridos en un archivo “Excel”.

El primer elemento que nos encontramos a la izquierda, es un editor de texto, guarda la ruta y el nombre de nuestro documento Excel. Este paso puede ser saltado, ya el botón “Guardar Excel” tiene la orden de guardar los datos en un archivo cuyo nombre lo tiene predeterminado por si no le queremos dar ningún nombre en especial. Siendo este nombre el que por defecto viene escrito en el editor, de manera que facilite el trabajo al usuario.

Si queremos ver tal archivo, podemos utilizar el siguiente elemento de nuestro panel de control, “abrir archivo Excel”, nos permitirá abrir este y otros archivos Excel.

La programación del editor del nombre del archivo es:

```
%Función editexcel: Esta función hace referencia al editor
"editexcel". Esta función guarda el nombre introducido en
el editor, este servirá para poner un nombre al archivo
nuevo
function editexcel_Callback (hObject, ~, handles)

     %El nombre introducido es transferido a la variable
          arcexcel = get(hObject, ‘String’);

     %El valor de la variable es guardado en memoria
     gracias al identificador
          handles.arcexcel = arcexcel;
          guidata (hObject, handles);

     %Guarda en nombre de la función
          guidata (editexcel_Callback);
```

La programación del botón “Guardar Archivo Excel” es:

```
%Función EXCEL: Esta función hace referencia al botón
"EXCEL". Esta función guarda todos los valores (Vmax, Frec,
Valores máx., medios y min.) obtenidos en el estudio en un
archivo Excel nombrado anteriormente
function EXCEL_Callback (~, ~, handles)

     %Muestra un mensaje en el panel de texto general
          Set (handles.text, ‘string’, [‘Los datos se han
          guardado en la ruta:’, handles.arcexcel])

     %Barra de espera
     %Abre o actualiza una ventana de diálogo de la barra
     de espera
          h_waitbar = waitbar (0, ‘Por favor, espere...’,
          ‘Position’,[337, 430, 270, 50],
          ‘Tag’,‘close_waitbar’,
          CloseRequestFcn’,...@close_waitbar_CloseRequestFc
          n);

     %Edita la barra de espera 300 veces
          for i = 1:300

          %Determina si la entrada es válida
               if ishandle (h_waitbar)
                    %Si es válida, repite la imagen e
                    incrementa la unidad(i)
                    waitbar (i / 300, h_waitbar)
               else
                    %Si no es válida, retorna al GUI
                    break
               end
          end

     %Cierra la ventana
           delete (h_waitbar)

     %Guarda en la variable "del" todos los datos: Vmax,
     frec., Valores máx., min. y medios
Del = {‘Circuito Cts.’, ‘Circuito Cts./Rampa’, ‘Circuito
Rampa’;
handles.arcexcelC1 handles.arcexcelR1 handles.arcexcelCR1;
handles.arcexcelC2 handles.arcexcelR2 handles.arcexcelCR2;
‘ ’, ‘ ’, ‘ ’;
‘Par máxima’, ‘Par mínimo’, ‘Par medio’;
handles.max1 handles.min1 handles.mean1;
handles.max2 handles.min2 handles.mean2;
handles.max3 handles.min3 handles.mean3;
```

```
'Velocidad máxima', 'Velocidad mínimo', 'Velocidad media';
handles.max12 handles.min12 handles.mean12;
handles.max22 handles.min22 handles.mean22;
handles.max32 handles.min32 handles.mean32;

'Ux máxima', 'Ux mínimo', 'Ux media';
handles.max13 handles.min13 handles.mean13;
handles.max23 handles.min23 handles.mean23;
handles.max33 handles.min33 handles.mean33;

'Uy máxima', 'Uy mínimo', 'Uy media';
handles.max14 handles.min14 handles.mean14;
handles.max24 handles.min24 handles.mean24;
handles.max34 handles.min34 handles.mean34};

%Escribe en el archivo Excel seleccionado, todos los datos
introducidos y obtenidos
    xlswrite (handles.arcexcel, del, 'Tensiones Y
    Frecuencias', 'F3')

%Guarda en nombre de la función
     guidata (EXCEL_Callback);
```

También nos encontramos el botón "info" el cual nos da información sobre este panel.

Panel de control: "Simulink".

Este último panel de control, mostrado en la figura 82, nos permite guardar el archivo Simulink en el caso de que lo hayamos variado y lo deseemos conservar. Lo que realmente hace es crear un archivo Simulink nuevo con un nombre diferente al existente. Esto lo podemos hacer nosotros escribiendo el nombre en el editor o dejar que el GUI lo haga, ya que por defecto utiliza el nombre "NewSimu". El archivo original volverá a sus valores iniciales cuando cerremos el GUI.

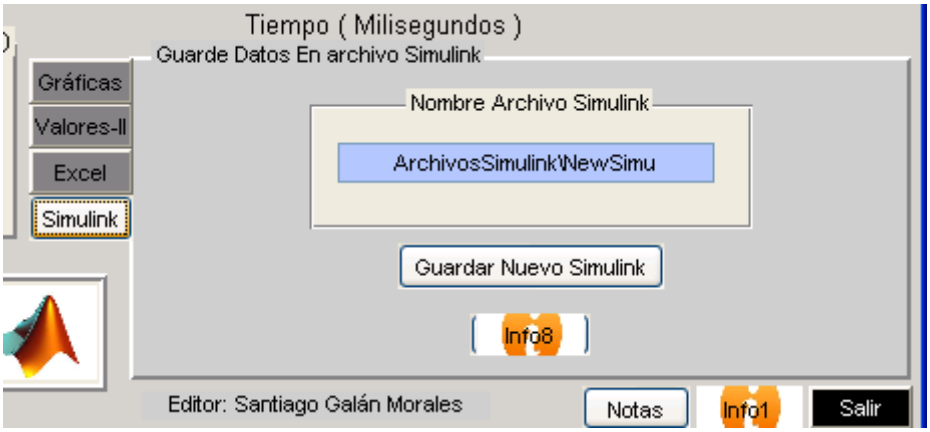

Figura 82. Simulink

La programación del editor del nombre del archivo es:

```
%Función editexcel: Esta función hace referencia al editor
"editexcel". Esta función guarda el nombre introducido en
el editor, este servirá para poner un nombre al archivo
nuevo
     function editexcel_Callback (hObject, ~, handles)

     %El nombre introducido es transferido a la variable
          arcexcel = get (hObject, 'String');

     %El valor de la variable es guardado en memoria
     gracias al identificador
          handles.arcexcel = arcexcel;
          guidata (hObject, handles);

     %Guarda en nombre de la función
          guidata (editexcel_Callback);
```

La programación del botón "guardar nuevo Simulink" es:

```
%Función EXCEL: Esta función hace referencia al botón
"EXCEL". Esta función guarda todos los valores (Vmax, Frec,
Valores máx., medios y min.) obtenido en el estudio en un
archivo Excel nombrado anteriormente
    function EXCEL_Callback (~, ~, handles)

    %Muestra un mensaje en el panel de texto general
          set (handles.text, 'string', ['Los datos se han
          guardado en la ruta:', handles.arcexcel])

    %Barra de espera
    %Abre o actualiza una ventana de diálogo de la barra
    de espera
          h_waitbar=waitbar(0,'Por favor, espere...',
          'Position', [337, 430, 270, 50], 'Tag',
          'close_waitbar','CloseRequestFcn',
          @close_waitbar_CloseRequestFcn);

     %Edita la barra de espera 300 veces
          for i = 1:300

          %Determina si la entrada es válida
               if ishandle (h_waitbar)

                    %Si es válida, repite la imagen e
                    incrementa la unidad(i)
                    waitbar (i / 300, h_waitbar)
                else
```

```
                    %Si no es válida, retorna al GUI
                    break
                end
            end

        %Cierra la ventana
            delete (h_waitbar)

        %Guarda en la variable "del" todos los datos: Vmax,
        frec., Valores máx., min. y medios
        Del = {‘Circuito Cts.’, ‘Circuito Cts./Rampa’,
        ‘Circuito Rampa’;
handles.arcexcelC1 handles.arcexcelR1 handles.arcexcelCR1;
handles.arcexcelC2 handles.arcexcelR2 handles.arcexcelCR2;

        ‘ ’,‘ ’,‘ ’;

  ‘Par máxima’, ‘Par mínimo’, ‘Par medio’;
     handles.max1 handles.min1 handles.mean1;
  handles.max2 handles.min2 handles.mean2;
  handles.max3 handles.min3 handles.mean3;

     ‘Velocidad máx’, ‘Velocidad mínimo’, ‘Velocidad
     media’;
  handles.max12 handles.min12 handles.mean12;
  handles.max22 handles.min22 handles.mean22;
  handles.max32 handles.min32 handles.mean32;

     ‘Ux máxima’,‘Ux mínimo’,‘Ux media’;
  handles.max13 handles.min13 handles.mean13;
  handles.max23 handles.min23 handles.mean23;
  handles.max33 handles.min33 handles.mean33;

     ‘Uy máxima’,‘Uy mínimo’,‘Uy media’;
     handles.max14 handles.min14 handles.mean14;
  handles.max24 handles.min24 handles.mean24;
  handles.max34 handles.min34 handles.mean34};

%Escribe en el archivo Excel seleccionado, todos los datos
introducidos y obtenidos
     xlswrite (handles.arcexcel, del, ‘Tensiones Y
     Frecuencias’, ‘F3’)

%Guarda en nombre de la función
     guidata (EXCEL_Callback);
```

También nos encontramos el botón “info” el cual nos da información sobre este panel.

5.5 FUNCIONES DE INICIO.

```
%Función de inicio 1. Esta función es escrita por Matlab
function varargout = PFCSGM (varargin) Gui_Singleton = 1;
Gui_State = struct (‘gui_Name’, mfilename, ‘gui_Singleton’,
gui_Singleton, ‘gui_OpeningFcn’, @PFCSGM_OpeningFcn,
‘gui_OutputFcn’, @PFCSGM_OutputFcn,’ gui_LayoutFcn’, [],
‘gui_Callback’, []);

     if nargin && ischar (varargin{1})
     Gui_State.gui_Callback = str2func (varargin {1});
     end

     if nargout
     [Varargout {1: nargout}] = Gui_mainfcn (gui_State,
     varargin {:});
     else
     Gui_mainfcn (gui_State, varargin{:});
     End

%Función de inicio 2: En ella se mencionan los valores
iniciales del programa
function PFCSGM_OpeningFcn (hObject, ~, handles, varargin)

     %Evita el sonido "beep" que sirve para avisar que se
     ha pulsado un botón, escrito algún comando, etc, que
     al cabo del tiempo resulta molesto
     beep off

     %Ocultamos el panel de control "Leyenda", ya que
     todavía no se ha requerido su utilización
     Set (handles.PanelLeyenda1, ‘Visible’, ‘off’);
     Set (handles.PanelLeyenda2, ‘Visible’, ‘off’);
     Set (handles.PanelLeyenda3, ‘Visible’, ‘off’);
     Set (handles.PanelLeyenda4, ‘Visible’, ‘off’);

     %Valor inicial a la variable "valcond", que nos ayuda
     para saber si está abierto el archivo Simulink
     "comparaciónUf"
     valcond = 0;
     handles.valcond = valcond;
     guidata (hObject, handles);

     %Centra el GUI (hay que poner las unidades del
     "propierty inspictor" en pixel)
     scrsz = get (0, ‘ScreenSize’);
     pos_act = get (gcf, ‘Position’);
     xr = scrsz(3)-pos_act(3);
     xp = round (xr/2);
     yr = scrsz(4)-pos_act(4);
```

```
yp = round(yr/2);
set (gcf,‘Position’,[xp yp pos_act(3)pos_act (4)]);

%Condiciones de inicio de los popmenu de l solver type
set (handles.uipanel55, ‘Visible’, ‘on’);
set (handles.uipanel56, ‘Visible’, ‘off’);

%Paneles de pestañas
%Botones: abrir, manipular, "introducirvalores"
%Muestra el panel principal y el resto los oculta
set(handles.panelvalores2, ‘Visible’, ‘on’);
set(handles.panelintroval, ‘Visible’, ‘off’);
set(handles.panelmanipulación, ‘Visible’, ‘off’);

%Destaca la pestaña seleccionada y el resto las
oscurece
set (handles.pestanaabrir, ‘backgroundcolor’,
[0.925 0.914 0.847]);
set (handles.pestanamanipular, ‘backgroundcolor‘,
[0.5 0.5 0.5]);
set (handles.pestanavalores1, ‘backgroundcolor’,
[0.5 0.5 0.5]);

%Botones: gráficas, valores y Excel
%Muestra el panel principal y el resto los oculta
Set (handles.panelgraficas, ‘Visible’, ‘on’);
Set (handles.panelvalores, ‘Visible’, ‘off’);
Set (handles.panelexcel, ‘Visible’, ‘off’);
Set (handles.pguaradrsimu, ‘Visible’, ‘off’);
Set (handles.panelmasgraficas, ‘Visible’, ‘off’);

%Destaca la pestaña seleccionada y el resto las
oscurece
Set (handles.pestanagraficas, ‘backgroundcolor’,
[0.925 0.914 0.847]);
Set (handles.pestanavalores2, ‘backgroundcolor’,
[0.5 0.5 0.5]);
Set (handles.pestanaexcel, ‘backgroundcolor’,
[0.5 0.5 0.5]);
Set (handles.pguardarSimu, ‘backgroundcolor’,
[0.5 0.5 0.5]);

%Se introduce los valores por defecto en los editores
vmax y frec en archivo "comparaciónUf"
handles.arcexcelC1 = 220/sqrt(3);
handles.arcexcelC2 = 2*pi*60;
handles.arcexcelR1 = (220/sqrt(3)-10)/0.75;
handles.arcexcelR2 = 120*pi/0.75;
handles.arcexcelCR1 = 220/sqrt(3);
handles.arcexcelCR2 = 120*pi/0.75;
guidata (hObject, handles);
```

```
%Introducimos el valor inicial del nombre del archivo
EXCEL por defecto
     %Guarda la palabra en la variable "a1"
     a1 = ‘Documentación/’;

     %Guarda la palabra en la variable "b1"
     b1 = ‘General (Guardar y Cargar)’;

     %Une las dos palabras
     c1 = strcat (a1, b1);

     %Guarda la variable
     handles.arcexcel = c1;
     guidata (hObject, handles);

%Introducimos el valor inicial del nombre del archivo
NEW SIMULINK por defecto

     %Guarda la palabra en la variable "a2"
     a2 = ‘ArchivosSimulink\’;

     %Guarda la palabra en la variable "b2"
     b2 = ‘NewSimu’;

     %Une las dos palabras
     c2 = strcat (a2, b2);

     %Guarda la variable
     handles.varsimu = c2;
     guidata (hObject, handles);

%Ingresa imagen de fondo
%Define en que panel de control irá la imagen
axes (handles.axes)

%Lee la imagen de archivo de gráficos
background1 = imread (‘imagenes/imagen.jpg’);

%Muestra la imagen
imshow (background1);

%Define en que panel de control irá la imagen
axes (handles.axes2);

%Lee la imagen de archivo de gráficos
background2 = imread (‘imagenes/imagen.jpg’);

%Muestra la imagen
imshow (background2);
%Define en que panel de control irá la imagen
axes (handles.axes3);
```

```
      %Lee la imagen de archivo de gráficos
      background3 = imread (‘imagenes/matlab.jpg’);

      %Muestra la imagen
      imshow (background3);

      %Valores iniciales de los sliders del archivo Simulink
      "comparaciónUf"
      set (handles.sliderC1, ‘Value’, 127);
      set (handles.sliderC2, ‘Value’, 477);
      set (handles.sliderR1, ‘Value’, 127);
      set (handles.sliderR2, ‘Value’, 250);
      set (handles.sliderCR1, ‘Value’, 127);
      set (handles.sliderCR2, ‘Value’, 250);


%COLOCA UNA IMAGEN EN CADA BOTÓN
      %Botón1
      %Lee la imagen de archivos gráficos
      [a] = imread (‘imagenes/abrir.bmp’);

      %Devuelve las dimensiones de la variable matricial
      [r, c] = size (a);

      %Redondea la cifra hacia el infinito positivo
      x = ceil (r/100);
      y = ceil (c/150);

      %Dimensiona la variable
      g = a (1:x:end, 1:y:end, :);
      g (g==255) = 5.5*255;

      %Aplica la imagen al botón mediante la variable
      set (handles.BotonAbrirArcSimu, ‘Cdata’, g);

      %Botón2
      [a] = imread (‘imagenes/play.bmp’);
      [r, c] = size(a)
      x = ceil (r/60);
      y = ceil (c/200);
      g = a (1:x:end, 1:y:end, :);
      g (g == 255) = 5.5*255;
      set (handles.BotonPlay1, ‘Cdata’, g);

      %Botón3
      [a] = imread (‘imagenes/play.bmp’);
      [r, c] = size (a);
      x = ceil (r/60);
      y = ceil (c/200);
      g = a (1:x:end, 1:y:end, :);
```

```
g (g==255) = 5.5*255;
set (handles.BotonPlay2, ‘Cdata’, g);

%Botón4
[a] = imread (‘imagenes/pdf.bmp’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.BotonAbrirDocu1, ‘Cdata’, g);

%Botón5
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.info1, ‘Cdata’, g);

%Botón6
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.info2, ‘Cdata’, g);

%Botón7
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.info3, ‘Cdata’, g);

%Botón8
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.info4, ‘Cdata’, g);

%Botón9
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
```

```
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.info5, ‘Cdata’, g);

%Botón10
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.info6, ‘Cdata’, g);

%Botón11
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a(1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.info7, ‘Cdata’, g);

%Botón12
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a(1:x:end, 1:y:end, :);
g (g==255) = 5.5*255;
set (handles.info8, ‘Cdata’, g);

%Botón13
[a] = imread (‘imagenes/info.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.info9, ‘Cdata’, g);

%Botón14
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.senalesp, ‘Cdata’, g);
```

```
%Botón15
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.senalesp, ‘Cdata’, g);

%Botón16
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.senalesUy, ‘Cdata’, g);

%Botón17
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a(1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.senalesUx, ‘Cdata’, g);

%Botón18
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.senalesv, ‘Cdata’, g);

%Botón19
[a] = imread (‘imagenes/señales.jpg’);
[r, c] = size (a);
x = ceil (r/30);
y = ceil (c/100);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.bParVel1, ‘Cdata’, g);

%Botón20
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/30);
y = ceil (c/100);
g = a (1:x:end, 1:y:end,:);
```

```
g (g==255) = 5.5*255;
set (handles.bParVel2, ‘Cdata’, g);

%Botó21
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/30);
y = ceil (c/100);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.bParVel3, ‘Cdata’, g);

%Botón22
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/30);
y = ceil (c/100);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.bUxUy1, ‘Cdata’, g);

%Botón23
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/30);
y = ceil (c/100);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.bUxUy2, ‘Cdata’, g);

%Botón24
[a] = imread (‘imagenes/senales.jpg’);
[r, c] = size (a);
x = ceil (r/30);
y = ceil (c/100);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.bUxUy3, ‘Cdata’, g);

%Botón25
[a] = imread (‘imagenes/abrir.bmp’);
[r, c] = size (a);
x = ceil (r/40);
y = ceil (c/110);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.Botonabrirall, ‘Cdata’, g);

%Botón26
[a] = imread (‘imagenes/abrir.bmp’);
[r, c] = size (a);
```

```
x = ceil (r/30);
y = ceil (c/100);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.BotonabrirArcComp, 'Cdata', g);

%Botón27
[a] = imread ('imagenes/borrar.jpg');
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end,:);
g (g==255) =5.5*255;
set (handles.BotonBorrar, 'Cdata', g);

%Botón28
[a] = imread ('imagenes/excel.jpg');
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.EXCEL, 'Cdata', g);

%Botón29
[a] = imread ('imagenes/abrir.bmp');
[r, c] = size (a);
x = ceil (r/60);
y = ceil (c/200);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.abrirexcel, 'Cdata', g);

%Botón 30
[a] = imread ('imagenes/abrir.bmp');
[r, c] = size (a);
x = ceil (r/100);
y = ceil (c/150);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.babrirsimu1, 'Cdata', g);

%Botón31
[a] = imread ('imagenes/play.bmp');
[r, c] = size (a);
x = ceil (r/45);
y = ceil (c/150);
g = a (1:x:end, 1:y:end,:);
g (g==255) = 5.5*255;
set (handles.bplay1, 'Cdata', g);
```

```
      %Botón32
      [a] = imread ('imagenes/pdf.bmp');
      [r, c] = size (a);
      x = ceil (r/45);
      y = ceil (c/150);
      g = a (1:x:end, 1:y:end,:);
      g (g==255) = 5.5*255;
      set (handles.babrirdocu1, 'Cdata', g);


%Muestra un mensaje de bienvenida en el panel de texto
general
      set (handles.text, 'String', 'BIENVENIDO A LA INTERFAZ
      GRÁFICA PFCSGM!!!');

%Guardamos los manejadores
      guidata (hObject, handles);




%Función de inicio 3
      function varargout = PFCSGM_OutputFcn (~,~,handles)
      varargout{1} = handles.output;
```

CAPÍTULO 6. MEJORAS.

MEJORA 1

Un dato importante es la redacción del código de programación de un GUI, ya que cuanto menor sea, más rápido se ejecutar sus acciones. Una manera de reducir la programación es evitar las variables locales y mencionar su configuración directamente.

El ejemplo práctico se muestra en la figura 83:

Figura 83. Ejemplo GUI.

Nos encontramos con una GUI que nos permite visualizar todo lo que vayamos escribiendo en el editor.

La programación es:

[Funciones de inicio]:
* * *
[Función del editor]:

```
%Función editor
function edit1_Callback (hObject, handles)

     %El valor introducido en el editor se ingresa en la
     variable
          var1 = get (hObject, 'string');

     %El contenido de la variable se guarda en el manejador
     o identificador
          handles.var1 = var1;

     %El contenido en el manejador "var1" se ingresa en el
     panel de texto llamado con el anejador text1
          set (handles.text1, 'string', handles.var1)
```

Esta sería una programación larga, la cual podría acortar de la siguiente manera:

[Funciones de inicio]:
* * *
[Función del editor]:

```
function edit1_Callback (hObject, eventdata, handles)

     %El dato introducido en el editor es ingresado
     directamente al panel de texto
          set (handles.text1, 'string',
          get(hObject,'string'))
```

MEJORA 2

En el panel gráfico de nuestro GUI no se puede hacer zoom, con lo que nos facilitaría el estudio de las señales. Para ello hay que sacar los boques "Scope" en el panel de control "Abrir", Sub-Panel "Modo Común". En estos bloques podemos manejar todo tipo de Zoom.

MEJORA 3

Con tal de mejorar la visualización del GUI, se podría haber realizado un GUI principal donde solo se podría apreciar un panel de control donde figurase botones que actuaran como enlaces a otros GUI ' s, estos, serían cada uno de los paneles de control de nuestro GUI.

MEJORA 4

Tambien podríamos haber creado un panel de control general donde nos permitiera guardar los datos (tanto los introducidos como los aportados) de cualquier archivo Simulink en un archivo Excel o en algún archivo de texto. Este GUI nos pernite hacerlo solo del archivo Simulink "comparaciónUf"

MEJORA 5

Una gran mejora, sería la comuncación con el exterior. El poder captar y mandar datos sería una gran herramienta para el control de motores, etc. Como así pasa con el software Labview capaz de controlar a través del puerto paralelo.

CAPÍTULO 7: CONCLUSIONES.

En este proyecto se ha desarrollado una interfaz gráfica de usuario para el manejo de modelos en Simulink de una forma más amigable. Se ha tratado de hacer una programación genérica adaptable a cualquier modelo de Simulink de una forma sencilla, aunque en este proyecto se ha particularizado para un modelo concreto, en el que se simulan distintos métodos de arranque de máquinas de inducción.

Con esta herramienta se pueden analizar, de una forma más dinámica, sencilla, gráfica e intuitiva, los resultados de la simulación de un modelo en Simulink, así como guardar las configuraciones más relevantes de un modelo en concreto para su posterior uso.En concreto, existen distintos modos de utilizar este GUI, uno más experimentado (Modo Experto) y otro un poco más genérico (Modo común).

Este proyecto puede servir de punto de partida para nuevos proyectos similares que contemplen una interacción genérica con cualquier modelo de Simulink y donde se incluyan las mejoras propuestas en este trabajo y otras muchas que se han dejado al margen para limitar el alcance del proyecto, muy extenso al tratarse de una herramienta muy completa donde el nivel de detalle lo marca la duración del proyecto y no las limitaciones del programa.

Me ha dado la oportunidad de aprender una herramienta, Matlab con sus aplicaciones más destacables como GUI y Simulink empezando desde cero, ya que no tenía noción alguna de esta herramienta de cálculo

También he abierto un pequeño camino ha futuros proyectos en esta universidad, ya que sobre este tema poco me he podido encontrar, dentro y fuera de ella. El futuro de este GUI puede estar como herramienta de apoyo para crear futuros GUI‘s centrados en otros sistemas Simulink o diversos sistemas de control, etc.

CAPÍTULO 8: BIBLIOGRAFÍA

-Ayuda del Matlab.

- [1] http: / / es.wikipedia.org / wiki / MATLAB
- [2] http: / / www.mathworks.es / products / simulink / index.html
- [3] “Control Of. Induction Motors”, Autor: Andrzej M. Trzynadlowski, Editorial: Academic Press
- [4] www.blinkdagger.com; blog de ingeniería y programación con Matlab.
- [5] “Matlab y sus Aplicaciones en las Ciencias y la Ingeniería”, Autor: César Pérez, Editorial: Pearson, Prentice Hall, 2002
- [6] “Matlab para ingenieros”, Autor: Jesus Fraile Ardanuy, Editorial: E.T.S.I.Caminos Canales, Madrid

ANEXOS

A. EXPLICACIÓN DE LOS COMPONENTES GRÁFICOS

GUI nos ofrece una paleta de herramientas donde nos podemos encontrar una serie de elementos gráficos. Estos componentes sirven para hacer una interfaz gráfica para el usuario. Estos elementos se posicionarán en un panel de control, este será un archivo con la extensión (*.fig)

Cada componente tendrá su función de referencia, donde se programará las acciones que realice cada vez que se ejecute ese componente. Las funciones se programarán en un archivo con la extensión (*.m). Tanto el nombre de la función y el nombre de componente serán iguales.

Por ejemplo:

Nombre del elemento: boton1

Nombre de la función: boton1_Callback

De aquí en adelante, se explicará de uno en uno todos los componentes utilizados en el proyecto. Se partirá de un índice y de la paleta de herramientas donde, visualmente, Matlab muestra los iconos de estos componentes para poder ser utilizados con facilidad.

Índice de los componentes utilizados en este proyecto

- Push Button, (Botón)
- Slider, (Deslizador)
- Radio Button, (Seleccionador)
- Check Box, (Verificador)
- Edit Text, (Editor)
- Static Text, (Panel de texto)
- Pop-Menú, (Menú de expansión)
- Listbox, (Menú)
- Toggle Button, (Interruptor)
- Table, (Tabla)
- Axes, (Panel gráfico)
- Panel, (Panel)
- ButtonGroup, (Panel agrupador)

La paleta de herramientas se muestra en la figura 84:

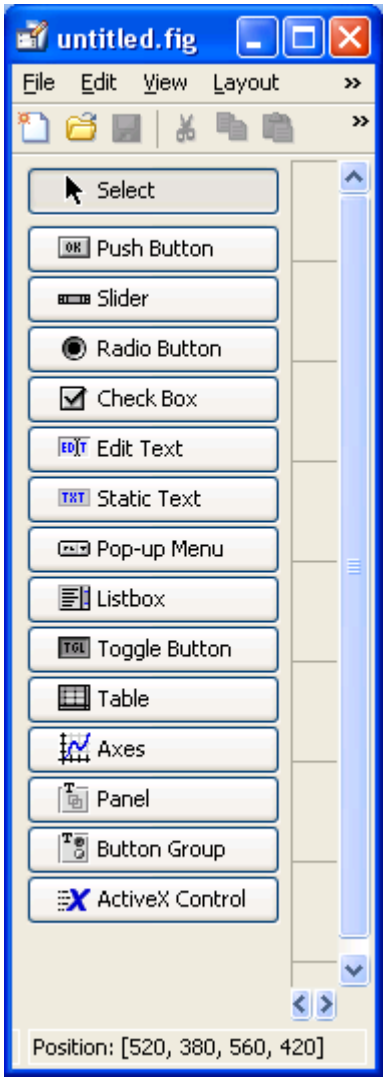

Figura 84. Paleta de herramientas.

PUSH BUTTON

Este componente es un simple botón.

Ejemplo de un ejercicio con un botón se muestra en la figura 85:

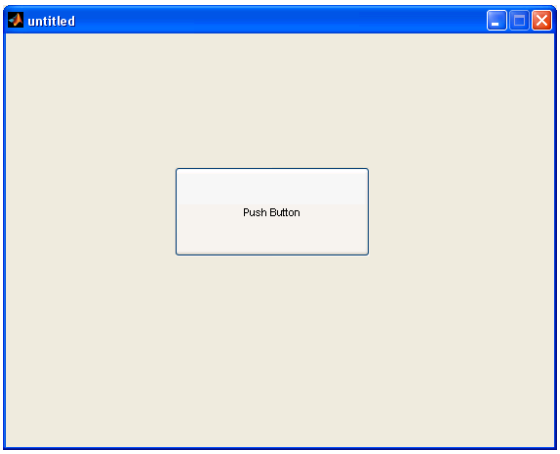

Figura 85. Push Button.

Este ejemplo contiene sólo un botón.
Si se hace un clic sobre el botón realiza la acción que contenga su función.

Por ejemplo:

Nombre de push button: boton1
Nombre de la función boton1_Callback

Archivo (*.fig)

```
%Se menciona la función
function boton1_Callback (~, ~, ~)

    %La acción de este botón es mostrar una ventana
    emergente donde se visualiza un texto
    Helpdlg ('TEXTO MOSTRADO', 'Ventana Info');
```

Este componente tiene propiedades que dan forma al componente para que se adapte a las necesidades del usuario.

SLIDER

Este componente es un objeto deslizante que determinar el valor de su variable de salida con un dispositivo deslizante dentro de su rango que va desde su posición inicial (valor mínimo) a su extremo opuesto (valor máximo).

Tanto su valor mínimo como su valor máximo, puede ser impuestos por el programador dentro de su propiedades.

Mostramos un ejemplo ilustrativo en la figura 86:

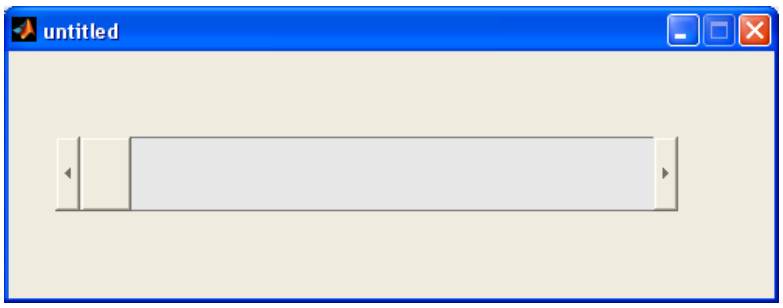

Figura 86. Slider (deslizador).

Ejemplo de un ejercicio con un slider:

- Archivo *.m
  Nombre de slider: slider1
- Archivo *.fig
  Nombre de la función slider1_Callback

```
% Se menciona la función
     function slider1_Callback (hObject, ~, handles)

% El valor del slider lo guardamos en la variable
     Variable = get (hObject, 'Value');

% La variable la guardamos en memoria
     Handles.variable = variable;
     Guidata (hObject, handles);
```

RADIO BUTTON

Este componente es un objeto gráfico, nos ayuda a mostrar de una manera gráfica, una selección.Es un objeto, que tras ser seleccionado, se queda en ese estado hasta que se deselecciona y cambia de estado. Esto es usado en la función como condición, de estar seleccionado hace una acción y de no estarlo haría otra acción.

Mostramos un ejemplo ilustrativo en la figura 87:

Figura 87. RadioButton.

Este ejemplo realiza tal prueba:

Archivo *.m
Nombre de slider: radiobutton1
Archivo *.fig
Nombre de la función radiobutton1_Callback

```
%Definimos la función del elemento “radiobutton”.
function radiobutton1_Callback (hObject,eventdata,handles)

    %Abrimos un condicionante para que según la selección
    hecha, hará una cosa u otra.
    if (get (hObject, ‘Value’) = get (hObject, ‘Max’))

    %Introduce un mensaje de texto al panel de control de
    texto.
    Set (handles.text, ‘el radio button ha sido
    seleccionado’, ‘string’);
    else

    %Introduce un mensaje de texto al panel de control de
    texto
    Set (handles.text, ‘el radio button ha sido
    deseleccionado’, ‘string’)
    end
```

También se puede cambiar el estado del “Radio-Button” de forma programable, colocando la propiedad “enable” en uno de sus valores máx. ó min.

El siguiente ejemplo ilustra la sintaxis posible para utilizar las propiedades:

```
%Oculta el objeto gráfico
Set (handles.radiobutton1, 'visible', off);
```

Sintaxis

```
Set(nombre del identificador del objeto,'propiedad',valor);
```

Existe una manera peculiar de utilizar un conjunto de varios objetos "radiobutton", si los agregamos en un panel de control, todos pueden ser seleccionados de manera independientes, si la programación no les ordena lo contrarío. Pero si los agregamos en un "Button-Group", todos los "Radio-Button" están relacionados y son dependientes, ya que solo uno podrá ser seleccionado a la vez y la selección de uno implica la no selección de otro.

CHECK BOX

Este componente es un objeto gráfico. En realidad, es una casilla de verificación. Donde se puede determinar el estado actual de una casilla de verificación seleccionándolo con el puntero del ratón o programándolo desde el editor de programación y utilizando la propiedad “value”.

Mostramos un ejemplo ilustrativo en la figura 88:

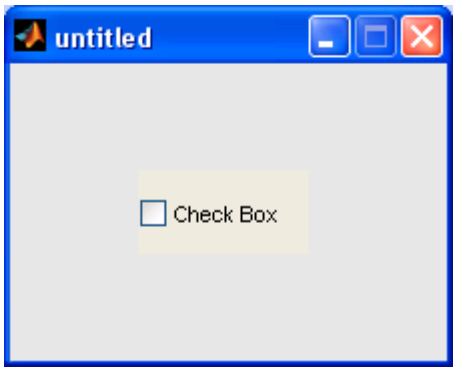

Figura 88. Check Box.

Este es un ejemplo ilustrado de tal prueba.

```
%Denominamos el nombre de la función del componente “Check-
Box”.
function checkbox1_Callback (hObject, eventdata, handles)

     %Condición
     If (get(hObject,‘Value’) = get(hObject,‘Max’))

     %El “Check-Box” está accionado
     Else

     %El “Check-Box” no está accionado
     end
```

EDIT TEXT

Este componente es un objeto editor. Este componente tiene la capacidad de aceptar del usuario entradas carácter de texto y de carácter numérico. Estas entradas, son guardadas en las variables. Las cuales son guardas con los identificadores para poder utilizar su memoria para su posterior uso. Uno de los usos es mostrar ese texto en paneles de control de texto, donde se visualiza el texto escrito:

Programación

```
%Nombramos la función del "editext"
function edit1_Callback (hObject, eventdata, handles)

     %Guardamos la entrada del usuario en la variable
     var1 = get (hObject, 'string');

     %Guardamos el contenido de la variable en memoria
     handles.var1 = var1;
     guidata (hObject, handles);

     %Nombramos la función del botón
     function boton1_Callback (hObject, ~, handles)

     %Mostramos en el panel de control de texto el
     contenido de la variable "variable"
     Set (handles.text1, 'string', handles.var1)
```

Demostración gráfica en la figura 89:

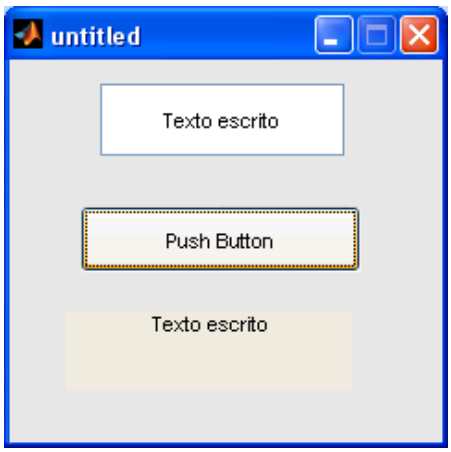

Figura 89. Edit text.

STATIC TEXT

Este componente es un objeto con la capacidad de mostrar el texto guardado en una variable.
Como ya hemos comentado en el objeto anterior, no habláremos más de “static-text”.

POP-UP MENÚ

Este componente es un objeto que tiene una capacidad de mostrarnos un menú desplegable de opciones entre las cuales podemos elegir.

Este objeto es un poco más complicado de programar que los anteriores, pero poco más, ya que al darnos varias opciones donde elegir, tendremos que dividir en partes nuestra programación.

Para ello utilizamos el condicionante “if ….else….end ”.

Demostración gráfica en la figura 90:

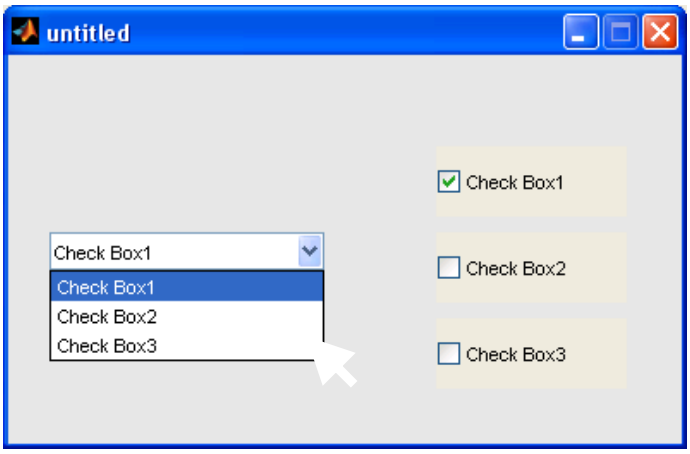

Figura 90. Pop-up menú.

Este guide, está compuesto por tres elementos “Check Box” y un elemento “pop-menú”, el cual nos da la capacidad de elegir entre tres opciones, entre seleccionar el “Check Box1”, “Check Box2” y “Check Box3”.Si seleccionamos la opción “Check Box1”, este se queda marcado.

La programación sería la siguiente:

```
%Se llama a la función.
function popupmenu1_Callback (hObject, ~, handles)

     %Se guarda la opción seleccionada en una variable.
     var1 = get (hObject, ‘Value’);

     %El comando switch determina que función será
     graficada.
     switch var1
          case 1
               set (handles.checkbox3, ‘value’, 1.0);
          case 2
               set (handles.checkbox2, ‘value’, 0.0);
          case 3
               set (handles.checkbox4, ‘value’, 0.0);
          end
```

```
%Se mencionan las funciones de los “Checbox”
function checkbox2_Callback(hObject,eventdata,handles)
function checkbox3_Callback(hObject,eventdata,handles)
function checkbox4_Callback(hObject,eventdata,handles)
```

LISTBOX

Este componente es un objeto que tiene una capacidad de mostrarnos un menú, no desplegable, de opciones entre las cuales podemos elegir. La manera de programar es muy parecida al componente mencionado con anterioridad, “Pop-menú”.

Demostración gráfica en la figura 91:

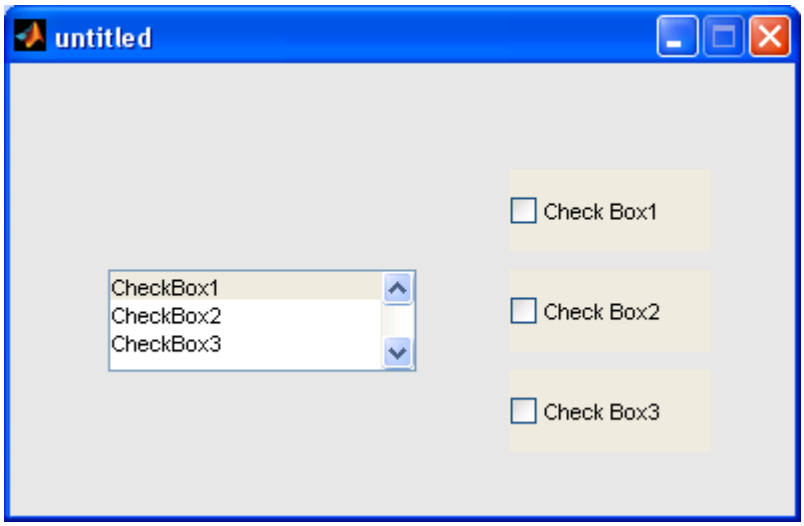

Figura 91. ListBox.

Este guide, está compuesto por tres elementos “Check Box” y un elemento “listbox”, el cual nos da la capacidad de elegir entre tres opciones, entre seleccionar el “Check Box1”, “Check Box2” y “Check Box3”. Si seleccionamos la opción “Check Box1”, este se queda marcado.

TOGGLE BUTTON

Este componente es un objeto igual que el “push-button”, pero tiene una pequeña diferencia. El “push button” es un pulsador, en cambio el “toglle button” es un interruptor.Esto le da la capacidad de tener doble capacidad de ejecución, es decir, cuando pulsamos podemos ejecutar una acción, y este botón se queda pulsado, al volver a pulsarlo, ejecuta otra opción y deja de estar pulsado.

Demostración gráfica en la figura 92:

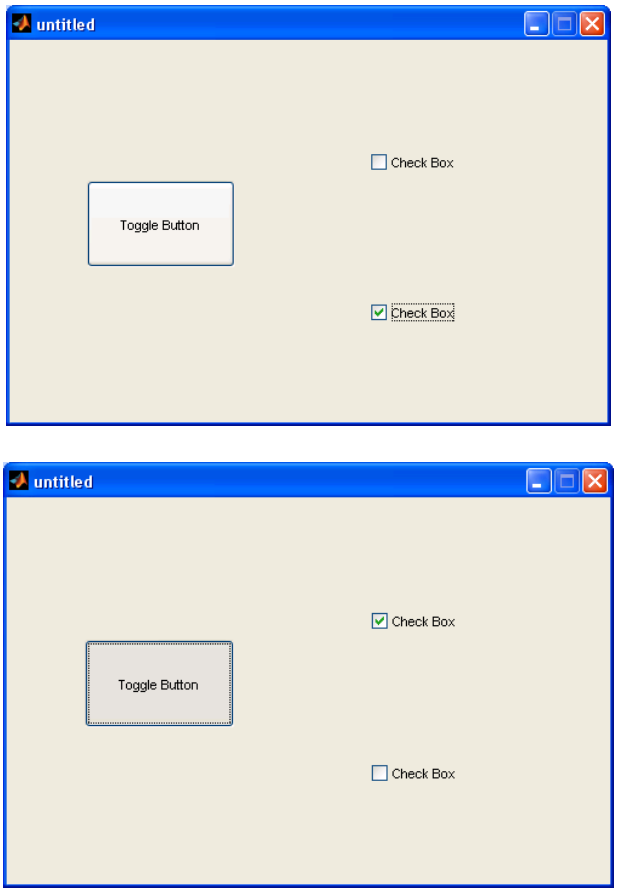

Figura 92. Toggle button.

En este ejemplo, vemos que cuando pulso, se activa el check box superior, cuando vuelvo a pulsar, se activa el check box inferior y se desactiva el check box superior.
Forma de programar:

```
%Definimos la función del elemento
Function togglebutton1_Callback (hObject,eventdata,handles)

     %Guardamos el valor del elemento
     button_state = get (hObject, ‘Value’);

     %Proponemos el condicionante.
     if button_state = get(hObject, ‘Max’);
```

```
%Si el Botón está pulsado, se activa el check box
superior
Set (handles.checkbox1, ‘value’, max)
elseif button_state = get (hObject, ‘Min’)

%Si el Botón no está pulsado, se activa el check box
inferior y se desactiva el check box superior.
Set (handles.checkbox1, ‘value’, min)
end
```

TABLE

Este componente puede contener caracteres de texto y numéricos así como elecciones preprogramadas (los menús desplegables). Cada columna debe contener el mismo tipo de datos. Se puede hacer una “table” dentro de otra. Se puede especificar formatos de la columna y de las filas. El número de filas y las columnas se ajustan automáticamente al tamaño de la matriz de datos.

Demostración gráfica en la figura 93:

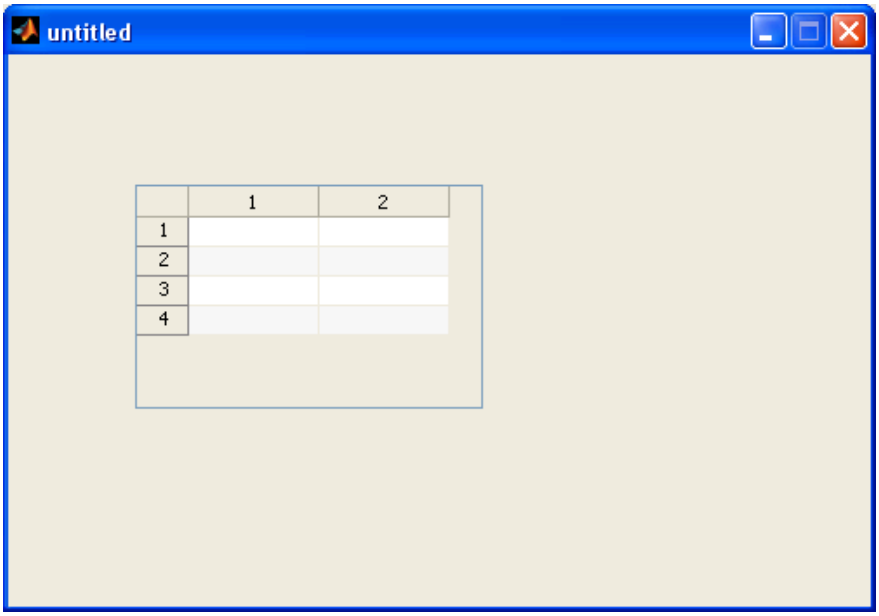

Figura 93. Table.

En este proyecto, no se utiliza

AXES

Este comando nos permite crear objetos gráficos en unos ejes. Los ejes crean un objeto gráfico en el cual se figuran objetos como señales, imágenes, etc. Matlab crea automáticamente unos ejes si no existieran.

Las propiedades de este comando, dan valores específicos a los ejes. (‘ PropertyName’, propertyvalue, ...) es la manera de redactar las propiedades con sus respectivos valore crea objetos teniendo en cuenta los valores especificados en la propiedad. Matlab destina valores predeterminados para cualquier propiedad de no ser nombrados.

Demostración gráfica en la figura 94:

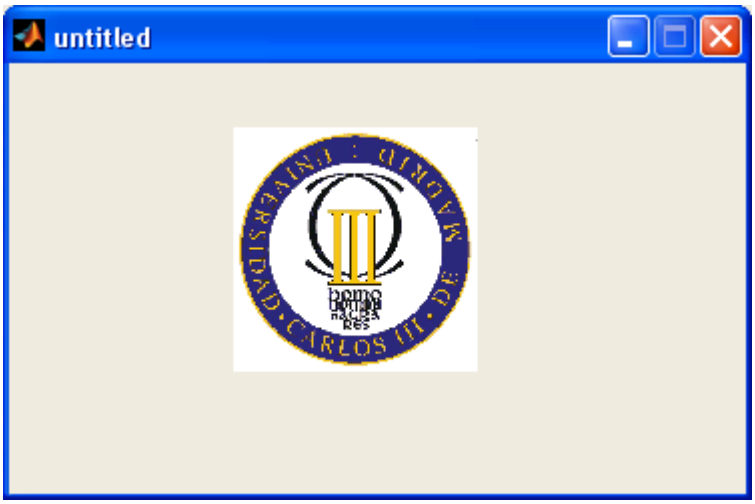

Figura 94. Axes.

Programación:

```
%Esta es la función que Matlab crea para definir los valores iniciales
function untitled_OpeningFcn (hObject, eventdata, handles, varargin)

    %Define en que panel de control irá la imagen
    axes (handles.axes1)

    %Lee la imagen de archivo de gráficos
    background1 = imread (‘imágenes/imagen.jpg’);

    %Muestra la imagen
    Imshow (background1);
```

PANEL

Este componente nos permite ordenar todos los componentes de nuestra interfaz gráfica y poder simplificar el guide, agrupando os objetos. Un panel puede contener botones, checkbox, paneles etc…Se puede controlar el tamaño y la posición del panel así como de sus componentes. Se puede hacer esto estableciendo el comportamiento de la Interfaz Gráfica del Usuario gracias a la propiedad de "Resize". No necesita programación, solo es gráfico.

Demostración gráfica en la figura 95:

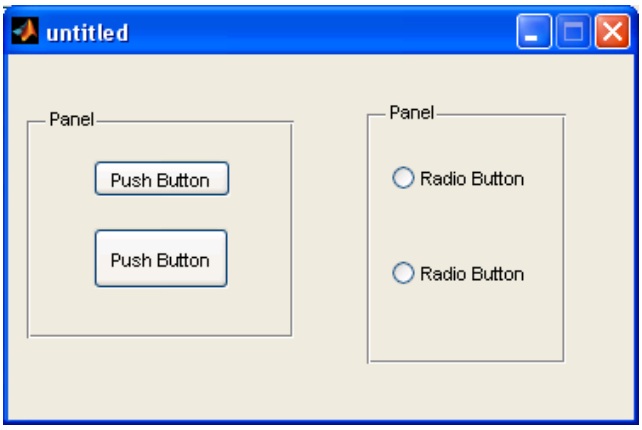

Figura 95. Panel.

BUTTON GROUP

Este componente es similar al componente anteriormente mencionado, “PANEL”. Pero tiene una peculiaridad que difiere del anterior. Esta diferencia es porque al agrupar varios elementos “check-box” sucede que tan solo uno puede ser seleccionado, cuando se seleccionado otro, este último queda deseleccionado. No necesita programación, solo es gráfico.

Demostración gráfica en la figura 96:

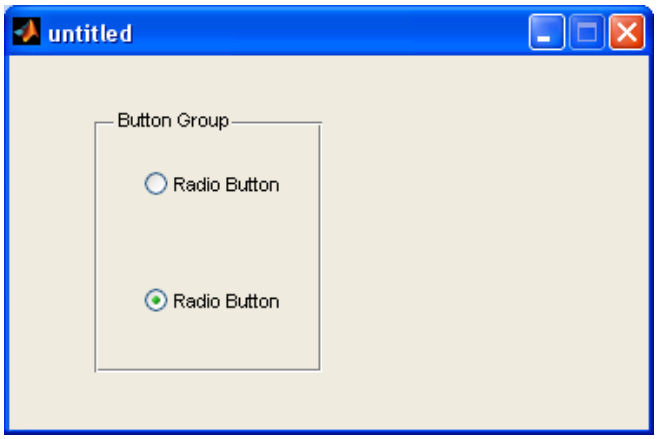

Figura 95. Button Group.

Principales propiedades de los componentes:

- **Tag**: Esta propiedad provee una manera para identificar objetos de gráficos con una etiqueta especificada por usuario. Esto es en particular útil cuando usted construye programas interactivos gráficos, que de otra manera necesitarían definir identificadores como variables globales.

- **Visible**: Esta propiedad determina si un objeto es exhibido en la pantalla. Si la propiedad está en “off”, el objeto es invisible.

- **Enable**: Esta propiedad deshabilita el botón, que aunque se vea, este no se puede ejecutar.

- **Position**: Esta propiedad especifica el tamaño y la posición en la pantalla del botón. Especifica el rectángulo de la posición con un vector de cuatro elementos de la forma:

  [eje x, eje y, la anchura, la altura]

- **Value:** Esta propiedad tiene la capacidad de seleccionar o deseleccionar el “Radio-Button” de manera que no haga falta utilizar el ratón para ello, si no de manera automática, referente a otra función que le de esa orden.

- **Backgroundcolor:** Esta propiedad tiene la capacidad de poner el color deseado al objeto gráfico.

Estas propiedades pueden ser cambiadas en el “Property inspector”. Se accede a él, desde el archivo (*.fig). También pueden ser cambiadas con la programación, la sentencia es:

```
%Inhabilita el botón
set (handles.botón1, ‘Enable’, ‘off’);
```

B. EXPLICACIÓN DE LOS DOCUMENTOS.

B.1 Introducción

En este apartado explicamos los diferentes documentos redactados donde podemos tomar información de nuestro GUI, así como información sobre Simulink y Matlab. A estos documentos podemos llegar si miramos dentro de las carpetas de este proyecto o desde nuestro GUI, gracias a los enlaces que nos muestra.

B.2. Explicación de los documentos

Nos encontramos con los documentos:

- 2.2.1. Parámetros archivo Simulink
- 2.2.2. Parámetros del bloque Constant
- 2.2.3. Parámetros del bloque Scope
- 2.2.4. Parámetros del bloque Ramp

B.3 Parámetros archivo Simulink

Este documento nos presenta los parámetros de la ventana de trabajo Simulink. Nos encontramos tres columnas. En la primera nos menciona el nombre del parámetro, que será la denominación para llamar al parámetro. En la segunda columna, nos encontramos la descripción, la cual nos ayuda a tomar información de las funciones que tiene dicho parámetro. Y por último, nos encontramos con la tercera columna, esta, nos informa de que tipo es el parámetro y de los valores que puede tomar este parámetro. Los valores que se encuentran entre llaves, {}, son los valores que Matlab pone por defecto de no ser nombrados estos parámetros. Los parámetros resaltados son los parámetros con mayor uso.

B.4 Parámetros del bloque constant

Este documento nos presenta los parámetros del bloque “constant”. Nos encontramos con la explicación de los parámetros que tiene este bloque.

B.5 Parámetros del bloque Scope

Este documento nos presenta los parámetros del bloque Scope. Nos encontramos dos columnas. En la primera nos menciona el nombre del parámetro, que será la denominación para llamar al parámetro. La segunda columna nos informa de que tipo es el parámetro y de los valores que puede tomar este parámetro. Los valores que se encuentran entre llaves, {}, son los valores que Matlab pone por defecto de no ser nombrados estos parámetros. Los parámetros resaltados son los parámetros con mayor uso.

B.6 Parámetros del bloque Ramp

Este documento nos presenta los parámetros del bloque “Ramp”. Nos encontramos con la explicación de los parámetros que tiene este bloque. (Slope, Start time…)